Deep Dynamic Neural Networks for Multimodal Gesture Segmentation and Recognition

Di Wu, Lionel Pigou, Ling Shao, Pieter-Jan Kindermans, and Joni Dambre

Abstract—This paper describes a novel method called Deep Dynamic Neural Networks*(DDNN)* for the dynamic multimodal gesture recognition. A generalised semi-supervised hierarchical dynamic framework is proposed for simultaneous gesture segmentation and recognition taking skeleton, depth and RGB images as input modules. Unlike traditionally constructing complex handcrafted features, all inputs modules are learnt by deep neural networks: the skeletal module is modeled by Deep Belief Networks*(DBN)*; the depth and RGB module are modeled by 3D Convolutional Neural Networks *(3DCNN)* to extract high level spatio-temporal features. Then the learned representations are used for estimating emission probabilities of the Hidden Markov Models to infer an action sequence. The framework can be easily extended by including an ergodic state to segment and recognise video sequences by a frame-to-frame mechanism, making online segmentation and recognition possible. This purely data-driven approach achieves ***0.81*** score in this gesture spotting challenge. The performance is on par with a variety of the state-of-the-art hand-tuned-feature approaches and other learning-based methods, opening the doors for using deep learning techniques to explore time series multimodal data.

Index Terms—Deep learning, 3D convolutional neural networks, sign language, segmentation and recognition.

1 Introduction

In recent years, human action recognition has drawn increasing attention of researchers, primarily due to its potential in areas such as video surveillance, robotics, human-computer interaction, user interface design, and multimedia video retrieval.

Previous works on video-based motion recognition focused on adapting handcrafted features and low-level hand-designed features. For example, [1], [2], [3] have been heavily employed with great success. These methods usually have two stages: an optional feature detection stage followed by a feature description stage. Well-known feature detection methods ("interest point detectors") are Harris3D [4], Cuboids [5] and Hessian3D [6]. For descriptors, popular methods are Cuboids [7], HOG/HOF [4], HOG3D [8] and Extended SURF [6]. In recent work of Wang *et al.* [9], dense trajectories with improved motion-based descriptors epitomized the pinnacle of handcrafted features and achieved state-of-the-art results on a variety of "in the wild" datasets. Based on the current trends, challenges and interests within the action recognition community, it is to be expected that many successes will follow. However the very high-dimensional dense-trajectory features usually require the use of advanced dimensionality reduction methods to make them computationally feasible.

Furthermore, as discussed in the evaluation paper of Wang *et al.* [10], there is no universally best hand engineered feature and the best performing method is dataset dependent. This clearly indicates that learning a dataset specific feature extractor, as is done in this work, can be highly beneficial. Even though hand-crafted features have dominated image recognition in previous years, there has been a growing interest in learning low-level and mid-level features, either in supervised, unsupervised, or semi-supervised settings [11], [12], [13].

The recent resurgence of neural networks invoked by Hinton and others [14], deep neural architectures serve as an effective solution for extracting high level features from data. Deep artificial neural networks (including the family of recurrent neural networks) have won numerous contests in pattern recognition and representation learning. Schmidhuber [15] compiled a historical survey compactly summarising relevant works with more than 850 entries of credited works. Such models have been successfully applied to a plethora of different domains: the GPU-based cuda-convnet [16] classifies 1.2 million high-resolution images into 1000 different classes; multi-column Deep Neural Networks [17] achieve near-human performance on the handwritten digits and traffic signs recognition benchmarks; 3D Convolutional Neural Networks [18], [19] recognize human actions in surveillance videos; Deep Belief Networks combining with Hidden Markov Models [20], [21] for acoustic and skeletal joints modeling outperform the decade-dominating paradigm of Gaussian Mixture Models+Hidden Markov Models. More recently, Baidu research proposed a DeepSpeech system [22] that combines a well-optimized RNN training system, achieving the best error rate on the noisy speech dataset. In these fields, deep architectures have shown great capacity to discover and extract higher level relevant features.

However, direct and unconstrained learning of complex problems remains difficult, since (i) the amount of required training data increases steeply with the complexity of the prediction model and (ii) training highly complex models with very general learning algorithms is extremely difficult. It is therefore common practice to restrain the complexity of the model and this is generally done by operating on small patches to reduce the input dimension and diversity [13], or by training the model in an unsupervised manner [12], or by forcing the model parameters to be identical for different input locations (as in convolutional neural networks [16], [17], [18]).

With the immense popularity of Kinect [23], [24], there has been renewed interest in developing methods for human gesture and action recognition from 3D skeletal data and depth images. A number of new datasets [25], [26], [27], [28] have provided researchers with the opportunity to design novel representations and algorithms and test them on a much larger number of sequences. It may seem that the task of action recognition based on 3D joint positions is trivial, but this is not the case, largely due to the high dimensionality and the huge amount of variability of the pose space. Furthermore, while the segmentation of the actions is as important as the recognition, it is an often neglected aspect of action recognition research.

in this paper we aim to address these issues by proposing a data driven system, focusing on analysis of acyclic video sequence labeling problems, *i.e.*, video sequences are non-repetitive as opposed to longer repetitive activities, *e.g.*, jogging, walking and running.

The key contributions of this work can be summarised as follows:

- We proposed a hierarchial dynamic framework that first extracts high level skeletal joint features and then used the learned representation for estimating emission probability to infer action sequences.
- We develop a 3D dynamic convolutional neural network architecture based on the convolution feature extractors for multiple channels inputs, *e.g.*, depth, grayscaled RGB with hand and body part as input. The proposed framework labels a video sequence in a frame-to-frame mechanism, rendering it possible for online segmentation and recognition for both RGB and depth images.
- We proposed a late fusion strategy for this dynamic hidden markov model, showing that multiple channel fusion outperforms individual module by a large margin.

2 Model Formulation

Inspired by the framework successfully applied to the speech recognition [20], the proposed model is a data driven learning system, relying on a pure learning approach. This results in an integrated model, where the amount of prior knowledge and engineering is minimised. On top of that, this approach works without the need for additional complicated pre-processing and dimensionality reduction methods.

2.1 Deep Dynamic Neural Networks

The proposed Deep Dynamic Neural Networks*(DDNN)* can be seen as an extension to [21], in that instead of only using the Restricted Boltzmann Machines to model human motion, various connectivity layers, *e.g.*, fully connected layers, convolutional layers, *etc.*, are stacked together to learn higher level features justified by a variational bound [14] from different input modules.

A continuous-observation HMM with discrete hidden states is adopted for modelling higher level temporal relationships. At each time step t, we have one observed random variable X_t. Additionally we have an unobserved variable H_t taking values of a finite set $\mathcal{H} = (\bigcup_{a\in\mathcal{A}} \mathcal{H}_a)$,where $\mathcal{H}_a$ is a set of states associated with an individual action $\boldsymbol{a}$ by force-alignment. The intuition motivating this construction is that an action is composed of a sequence of poses where the relative duration of each pose may vary. This variance is captured by allowing flexible forward transitions within the chain. With this definitions, the full probabilistic model is now specified as a Hidden Markov Model:

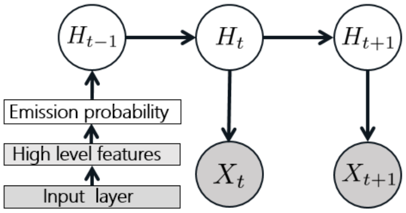


Fig. 1: Per-action model: a forward-linked chain. Inputs (skeletal features, or depth & RGB image features) are first passed through Deep Neural Nets (Deep Belief Networks for skeletal modality or 3D Convolutional Neural Networks for depth & RGB modality) to extract high level features. The outputs are the emission probabilities of the hidden states.

$$p(H_{1:T}, X_{1:T}) = p(H_1)p(X_1|H_1)\prod_{t=2}^{T} p(X_t|H_t)p(H_t|H_{t-1}), \quad (1)$$

where $p(H_1)$ is the prior on the first hidden state; $p(H_t|H_{t-1})$ is the transition dynamics model and $p(X_t|H_t)$ is the emission probability modelled by the deep neural nets.

The motivation for using deep neural nets to model the emission probabilities conditional distributions is that by constructing multi-layer networks, semantically meaningful high level features will be extracted whilst learning the parametric prior of human pose from mass pool of data. In the recent work of [29], a non-parametric bayesian network is adopted for human pose prior estimation, whereas in the proposed framework, the parametric networks are incorporated. The graphical representation of a per-action model is shown as Fig. 1.

2.2 Ergodic States Hidden Markov Model

The aforementioned framework can be easily adapted for simultaneous action segmentation and recognition by adding an ergodic state ($\mathcal{ES}$) which resembles the silence state for speech recognition. Hence, the hidden variable H_t can take on an extra value within the finite set, which becomes $\mathcal{H} = (\bigcup_{a\in\mathcal{A}} \mathcal{H}_a) \bigcup \mathcal{ES}$, where $\mathcal{ES}$ is the ergodic state as the resting position between actions. We refer the model as Ergodic States Hidden Markov Model (*ES-HMM*) for simultaneously gesture segmentation and recognition.

Since our goal is to capture the variation in speed of the performed gestures, we set the transitions matrix $\mathbf{p(H_t|H_{t-1})}$ in the following way as shown in Fig. 2 : when being in a particular node n in time t, moving to time $t+1$, we can either stay in the same node (slower performance), move to node $n+1$ (the same speed of performance), or move to node $n+2$ (faster performance). From the $\mathcal{ES}$ we

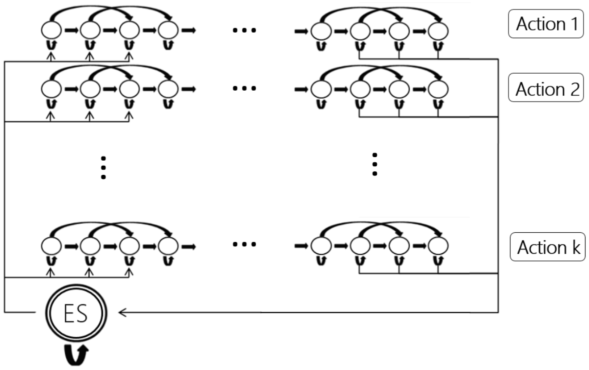


Fig. 2: State diagram of the *ES-HMM* model for low-latency action segmentation and recognition. An ergodic states (ES) shows the resting position between action sequence. Each node represents a single frame and each row represents a single action model. The arrows indicate possible transitions between states.

can move to the first three nodes of any gesture class, and from the last three nodes of any gesture class we can move to the $\mathcal{ES}$.

The *ES-HMM* framework differs from the Firing Hidden Markov Model of [30] in that we strictly follow the temporal independent assumption, forbidding inter-states transverse, preconditioned that a non-repetitive sequence would maintain its unique states throughout its performing cycle.

The emission probability of the trained model is represented as a matrix of size $N_{\mathcal{TC}} \times N_{\mathcal{F}}$ where $N_{\mathcal{F}}$ is the number of frames in a test sequence and output target class $N_{\mathcal{TC}} = N_{\mathcal{A}} \times N_{\mathcal{H}_a} + 1$ where $N_{\mathcal{A}}$ is the number of action class and $N_{\mathcal{H}_a}$ is the number of states associated to an individual action a and one $\mathcal{ES}$ state (*c.f.*, Fig. 12: x-axis as $N_{\mathcal{F}}$ and y-axis as $N_{\mathcal{TC}}$ with $\mathcal{ES}$ as the bottom y-axis 101).

Once we have the trained model, we can use the normal online or offline smoothing, inferring the conditional distributions $p(H_t|X_t)$ of every node (frame) of the test video. Because the graph for the Hidden Markov Model is a directed tree, this problem can be solved exactly and efficiently using the max-sum algorithm. The number of possible paths through the lattice grows exponentially with the length of the chain. The Viterbi algorithm searches this space of paths efficiently to find the most probable path with a computational cost that grows only linearly with the length of the chain [31]. We can infer the action presence in a new sequence by Viterbi decoding as:

$$V_{t,\mathcal{H}} = \log P(H_t|X_t) + \log(\max_{\mathcal{H} \in \mathcal{H}_a}(V_{t-1,\mathcal{H}})) \tag{2}$$

where initial state $V_{1,\mathcal{H}} = P(H_1|X_1)$. From the inference results, we define the probability of an action $a \in \mathcal{A}$ as $p(y_t = a|x_{1:t}) = V_{T,\mathcal{H}}$. Result of the Viterbi algorithm is a path–sequence of nodes which correspond to hidden states of gesture classes. From this path we can infer the class of the gesture (*c.f.*, Fig. 12). The overall algorithm for training and testing are presented in Algorithm 1 and 2 .

Algorithm 1: Multimodal Deep Dynamic Networks – training

```
Data:
$\mathbf{X^1} = \{\mathbf{x_i^1}\}_{\mathbf{i} \in [\mathbf{1}\ldots\mathbf{t}]}$ - raw input(skeletal) feature
    sequence.
$\mathbf{X^2} = \{\mathbf{x_i^2}\}_{\mathbf{i} \in [\mathbf{1}\ldots\mathbf{t}]}$ - raw input(depth) feature sequence
    in the form of $M_1 \times M_2 \times T$, where $M_1, M_2$ are
    the height and width of the input image and $T$
    is the number of contiguous frames of the
    spatio-temporal cuboid.
$\mathbf{Y} = \{\mathbf{y_i}\}_{\mathbf{i} \in [\mathbf{1}\ldots\mathbf{t}]}$ - frame based local label (achieved by
    semi-supervised forced-aligment), where
    $\mathbf{y_i} \in \{C * S + \mathit{1}\}$ with $C$ is the number of class,
    $S$ is the number of hidden states for each class,
    $\mathit{1}$ as ergodic state.
for $m \leftarrow 1$ to $2$ do
    if $m$ is $1$ then
        Preprocess skeletal data $\mathbf{X^1}$ as in Eq.3 4 5.
        Normalize(zero mean, unit variance per
        dimension) the above features and feed to to
        Eq.6.
        Pre-train the networks using Contrastive
        Divergence.
        Supervised fine-tuning of the Deep Belief
        Networks using $\mathbf{Y}$ by standard mini-batch
        SGD backpropagation.
    else
        Preprocess the depth and RGB data $\mathbf{X^2}$ as in
        3.2.1.
        Feed the above features to Eq.9.
        Train the 3D Convolutional Neural Networks
        using $\mathbf{Y}$.
Result:
GDBN - a gaussian bernoulli visible layer Deep
    Belief Network to generate the emission
    probabilities for hidden markov model.
3DCNN - a 3D Deep Convolutional Neural
    Networks to generate the emission probabilities
    for hidden markov model.
$\mathbf{p(H_1)}$ - prior probability for $\mathbf{Y}$ by accumulating and
    normalizing labels.
$\mathbf{p(H_t|H_{t-1})}$ - transition probability for $\mathbf{Y}$, enforcing
    the beginning and ending of a sequence can
    only start from the first or the last state.
```

3 Model Implementation

In all our experiments the number of states associated to an individual action $N_{\mathcal{H}_a}$ is chosen as 5 for modeling the states of an action class, therefore $N_{\mathcal{TC}} = 20 \times 5 + 1 = 101$. The labels for each cuboid **Y** are specified as follows:

Hidden states($\mathcal{H}_a$): Force alignment is used to extract the hidden states, *i.e.*, if a gesture token is 100 frames, the first 10 frames are assigned as hidden state ***1*** and the 10-20 frames are assigned as hidden state **2** and so on and so forth.

Ergodic states($\mathcal{ES}$): Neutral frames are extracted as 5 frames before or after a gesture tokens labelled by ground truth.

Algorithm 2: Multimodal Deep Dynamic Networks – testing

```
Data:
$\mathbf{X^1} = \{\mathbf{x_i^1}\}_{\mathbf{i}\in[\mathbf{1}\dots\mathbf{t}]}$ - raw input(skeletal) feature
        sequence.
$\mathbf{X^2} = \{\mathbf{x_i^2}\}_{\mathbf{i}\in[\mathbf{1}\dots\mathbf{t}]}$ - raw input(depth) feature sequence
        in the form of $M \times M \times T$.
GDBN - trained gaussian bernoulli visible layer
        Deep Belief Network to generate the emission
        probabilities for hidden markov model.
3DCNN - trained 3D Deep Convolutional Neural
        Networks to generate the emission
        probabilities for hidden markov model.
$\mathbf{p(H_1)}$ - prior probability for $\mathbf{Y}$.
$\mathbf{p(H_t|H_{t-1})}$ - transition probability for $\mathbf{Y}$.
for $m \leftarrow 1$ to $2$ do
    if $m$ is $1$ then
        Preprocessing and normalizing the data $\mathbf{X^1}$ as
        in 3 4 5.
        Feedforwarding network GDBN to generate
        the emission probability $\mathbf{p(X_t|H_t)}$ in Eq.1.
        Generating the score probability matrix
        $\mathbf{S^1} = \mathbf{p(H_{1:T}, X_{1:T})}$.
    else
        Preprocessing the data $\mathbf{X^2}$ (normalizing,
        median filtering the depth data).
        Feedforwarding 3DCNN to generate the
        emission probability $\mathbf{S^2} = \mathbf{p(X_t|H_t)}$ in Eq.1.
        Generating the score probability matrix
        $\mathbf{S^2} = \mathbf{p(H_{1:T}, X_{1:T})}$.
Fuse the score matrix $\mathbf{S} = \alpha * \mathbf{S^1} + (\mathbf{1} - \alpha) * \mathbf{S^2}$.
Finding the best path $\mathbf{V_{t,\mathcal{H}}}$ using $\mathbf{S}$ by Viterbi
decoding as in Eq.2.
Result:
$\mathbf{Y} = \{\mathbf{y_i}\}_{\mathbf{i}\in[\mathbf{1}\dots\mathbf{t}]}$ - frame based local label
        where $\mathbf{y_i} \in \{C * S + \mathit{1}\}$ with $C$ is the number
        of class, $S$ is the number of hidden states for
        each class, $\mathit{1}$ as ergodic state.
C - global label, the anchor point is chosen as the
        middle state frame.
```

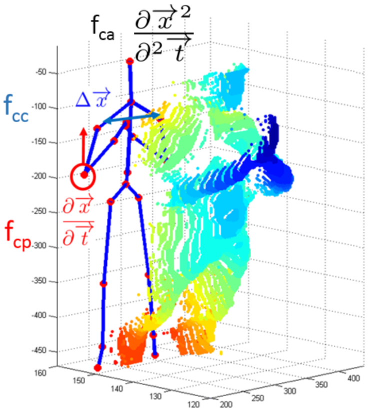

f_ca
$\frac{\partial \overrightarrow{x}^2}{\partial^2 \overrightarrow{t}}$
f_cc
$\Delta \overrightarrow{x}$
f_cp
$\frac{\partial \overrightarrow{x}}{\partial \overrightarrow{t}}$
-50
-100
-150
-200
-250
-300
-350
-400
-450
160
150
140
130
120
200
250
300
350
400

Fig. 3: Point cloud projection of depth image and the 3D positional features.

3.1 Skeleton Module

3.1.1 Preprocessing

Only upper body joints are relevant to the discriminative gesture recognition tasks. Therefore, only the 11 upper body joints are considered. The 11 upper body joints used are *"ElbowLeft, WristLeft, ShoulderLeft, HandLeft, ElbowRight, WristRight, ShoulderRight, HandRight, Head, Spine, HipCenter"*.

The 3D coordinates of the N joints of frame c are given as: $X_c = \{x_1^c, x_2^c, \dots, x_N^c\}$. 3D positional pairwise differences of joints [21] are used in the representation of the observed variable $\mathcal{X}$. They capture posture features, motion features by direction concatenation: $\mathcal{X} = [f_{cc}, f_{cp}, f_{ca}]$ as in Eq. 3 4 5. Note that offset features used in [21] depend on the first frame, if the initialization fails which is a very common scenario, the feature descriptor will be generally very noisy. Hence, the offset features are discarded and only the three more robust features $[f_{cc}, f_{cp}, f_{ca}]$ (as shown in Fig. 3) are kept for representing the frame pairwise difference, velocity, acceleration elements for skeletal features:

$$f_{cc} = \{x_i^c - x_j^c | i, j = 1, 2, \dots, N; i \neq j\} \quad (3)$$

$$f_{cp} = \{x_i^c - x_i^p | x_i^c \in X_c; x_i^p \in X_p\} \quad (4)$$

$$f_{ca} = \{x_i^p - 2 \times x_i^c + x_i^n | x_i^c \in X_c; x_i^p \in X_p; x_i^n \in X_n\} \quad (5)$$

with X_i^c, X_i^p, X_i^n as the current, previous, next frame skeletal features.

This results in a raw dimension of $N_{\mathcal{X}} = N_{joints} * (\frac{N_{joints}}{2} + N_{joints} + N_{joints}) * \mathit{3}$ where N_{joints} is the number of joints used. Therefore, in the experiment with $N_{joints} = 11, N_{\mathcal{X}} = 891$. Admittedly, we do not completely neglect human prior knowledge about information extraction for relevant static postures, velocity and acceleration of overall dynamics of motion data. While we have indeed used prior knowledge about the relevant features, the resulting ones remain quite general and do not need data-set specific tuning. A similar data driven approach has been adopted in [26] where random forest classifiers were adapted to the problem of recognizing gestures using a bundle of 35 frames. These sets of feature extraction processes resemble the *Mel Frequency Cepstral Coefficients (MFCCs)* for the speech recognition community [20].

3.1.2 Gaussian Bernoulli Restricted Boltzmann machines

Because input skeletal features(*a.k.a.*observation domain $\mathcal{X}$) are continuous instead of binomial features, we use the Gaussian RBM (*GRBM*) to model the energy term of first visible layer:

$$E(v, h; \theta) = -\sum_{i=1}^{D} \frac{(v_i - b_i)^2}{2\sigma_i^2} - \sum_{i=1}^{D}\sum_{j=1}^{F} W_{ij} h_j \frac{v_i}{\sigma_i} - \sum_{j=1}^{F} a_j h_j \quad (6)$$

The conditional distributions needed for inference and generation are given by:

$$P(h_{j=1}|\mathbf{v}) = g(\sum_i W_{ij} v_i + a_j)); \quad (7)$$

$$P(v_{i=1}|\mathbf{h}) = \mathcal{N}(v_i|\mu_i, \sigma_i^2). \quad (8)$$

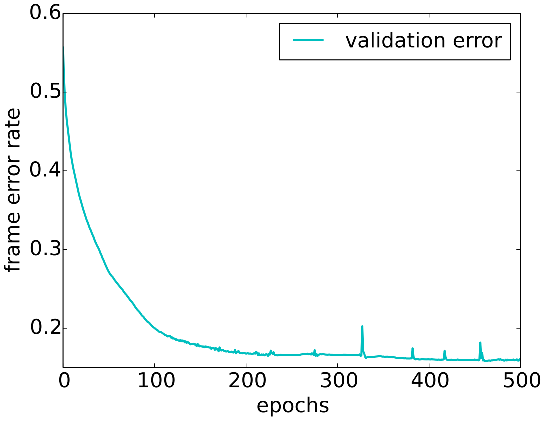


Fig. 4: Deep Belief Networks training frame based validation set error rate for skeleton input module.

where $\mu_i = b_i + \sigma_i^2 \sum_j W_{ij} h_j$ and $\mathscr{N}$ is normal distribution. In general, we normalize the data (mean substraction and standard deviation division) in the preprocessing phase. Hence, in practice, instead of learning σ_i^2, one would typically use a fixed, predetermined unit value ***1*** for σ_i^2.

For high level skeleton feature extraction, the network architectures is $[N_{\mathcal{X}}, 2000, 2000, 1000, N_{\mathcal{TC}}]$, where $N_{\mathcal{X}} = 891$ is the observation domain dimension; $N_{\mathcal{TC}}$ is the output target class.

3.1.3 Deep Belief Networks Pretraining & Training Details

In the training set, there are in total $400,117$ frames. During the training of *DBN*, 90% is used for training, 8% for validation (for the purpose of early stopping) 2% is used for test evaluation. The feed forward networks are pre-trained with a fixed recipe using stochastic gradient decent with a mini-batch size of 200 training cases. Unsupervised initializations tend to avoid local minima and increase the networks performance stability and we have run 100 epochs for unsupervised pre-training. For Gaussian-binary RBMs, learning rate is fixed at 0.001 while for binary-binary RBMs as 0.01 (note in generally training *GRBM* requires smaller learning rate). For fine-tuning, the learning rate starts at 1 with 0.99999 mini-batch scaling. Maximum number of fine-tuning epoch is 500 with early stopping strategy and during the experiments, early stopping occurs around 440 epoch. Optimization complete with best validation score (the frame based prediction error rate) of 16.5%, with test performance 16.15%. The frame based validation error rate is shown as Fig 4 .

Though we believe further carefully choosing network architecture would lead to more competitive results, in order not to "creeping overfitting", as algorithms over time become too adapted to the dataset, essentially memorizing all its idiosyncrasies, and losing ability to generalize [32], we would like to treat the model as the aforementioned more generic approach. Since a completely new approach will initially have a hard time competing against established, carefully fine-tuned methods. The performance of skeleton module is shown in Tab 1.

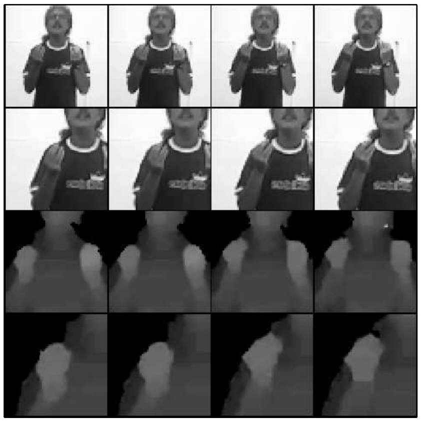


Fig. 5: Preprocessing steps. Inputs from top to bottom: 1) gray-scale body input, 2) gray-scale hand input, 3) depth body input, 4) depth hand input.

3.2 RGB & Depth 3D Module

3.2.1 Preprocessing

Working directly with raw input Kinect recorded data frames, which are 480×640 pixel images, can be computationally demanding. Deepmind technology [33] presented the first deep learning model to successfully learn control policies directly from high-dimensional sensory input using deep reinforcement learning.

Our first step in the preprocessing stage is cropping the highest hand and the upper body using the given joint information. We discovered that the highest hand is the most interesting. If both hands are used, they perform the same (mirrored) movement. If one hand is used, it is always the highest one. If the left hand is used, the videos are mirrored. This way, the model only needs to learn one side.

The preprocessing results in four video samples (body and hand with gray-scale and depth) of resolution 4x64x64 (4 frames of size 64x64). Furthermore,the noise in the depth maps is reduced with thresholding, background removal using the user index, and median filtering. The outcome is shown in Figure 5.

3.2.2 3DCNN Architecture

The 3D convolution is achieved by convolving a 3D kernel to the cuboid formed by stacking multiple contiguous frames together. We follow the nomenclature as in [19]. However, instead of using $tanh$ unit as in [19], the Rectified Linear Units (*ReLUs*) [16] were adopted where trainings are several times faster than their equivalents with $tanh$ units. Formally, the value of a unit at position (x, y, z) (z here corresponds the time-axis) in the jth feature map in the ith layer, denoted as v_{ij}^{xyz}, is given by:

$$v_{ij}^{xyz} = max(0, (b_{ij} + \sum_m \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} \sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(t+r)})) \quad (9)$$

The 3DCNN architecture is depicted as Fig. 6 : the 4 types 5 of input contextual frames are stacked as size

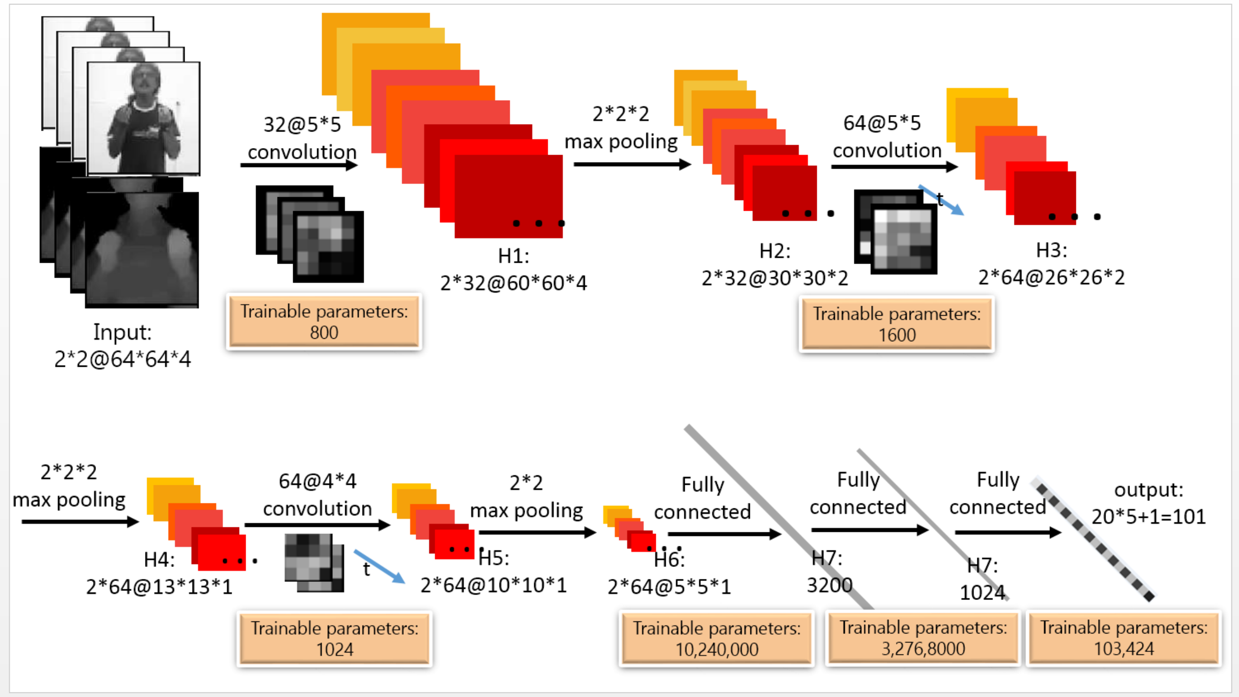


Fig. 6: An illustration of the architecture of the 3DCNN architecture.

$64 \times 64 \times 4$. The depth images are normalized with N_{var} and the grayscale images are normalized with N_{std} as in 10,11 because the median of depth images is irrelevant to the gesture subclass but the grayscale images is.

$$N_{var} = (x - \boldsymbol{mean}(x))/(\boldsymbol{var}(x))^{1/2} \tag{10}$$

$$N_{std} = x/(\boldsymbol{var}(x))^{1/2} \tag{11}$$

The first layer contains 32 maps of 5×5 kernel followed by local contrast normalization (LCN) layer [34] and stride (2,2,2) max pooling, then the grayscale channel and depth channel are concatenated; the second convolutional layers has 64 maps of 5×5 kernel followed by LCN layer and stride (2,2,2) max pooling; the third convolution layer is composed of 64 maps of 4×4 kernel followed by stride (1,2,2) max pooling; then all convnets outputs are flattened with the body channel and hand channel concatenated into one fully connected layer of size 1024; the output layer $N_{\mathcal{TC}}$ is of size $101 = 5 \times 20 + 1$ (number of hidden states for each class× number of classes plus one ergodic state).

3.2.3 Details of Learning

The first 650 batches are used for training and the remaining 50 files for validation. During training, dropout [35] is used as main regularization approach to reduce overfitting. Nesterovs accelerated gradient descent (NAG) [36] with a fixed momentum-coefficient of 0.9 and mini-batches of size 64 are also used. The learning rate is initialized at 0.003 with a 5% decrease after each epoch. The weights of the CNNs are randomly initialized with a normal distribution with $\mu = 0$ and $\sigma = 0.04$. The frame-based classification error ran for 40 epochs is 39.062 as validation error is shown in Fig. 7. Comparing with the skeleton module frame based error rate in Fig 4 , it can be seen that skeleton module has lower frame based error rate.

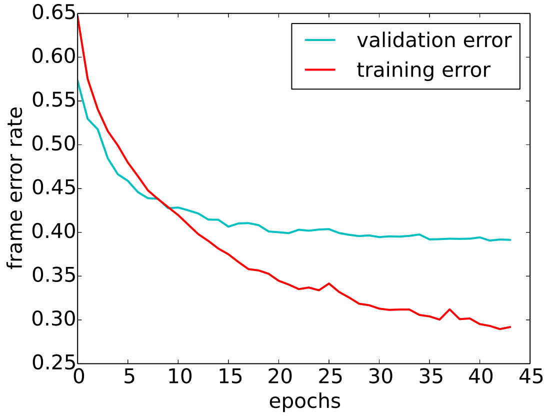


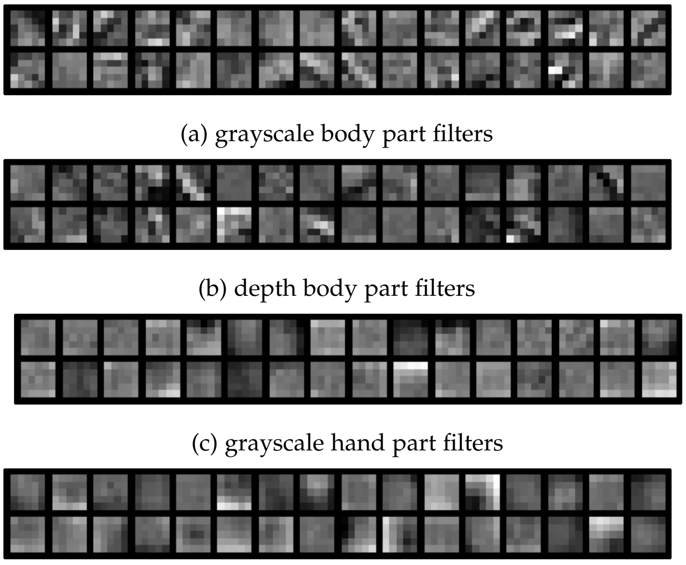
Fig. 7: The frame-based error rate for training 3DCNN.

3.2.4 Looking into the networks: visualization of filter banks

The weight filters of the first *conv1* layer are depicted in Fig. 8 and it can be seen the unique characteristics from the filter kernels. For body and hand filters, both inputs are of the same size : 64×64. Hand inputs are smoother than the body inputs, hence the hand part filters are smoother than the body part filters. For depth and grayscale image filters: depth images generally have less edges/smoother. Hence, depth filters are smoother than the grayscale filters (though the distinctions are less obvious compared with the body vs. hand part filters).

3.3 Post-Processing

The predicted token less than 20 frames are discarded as noisy tokens. Note that there are many noisy gesture tokens predicted by viterbi decoding. One way to sift through the



(a) grayscale body part filters

(b) depth body part filters

(c) grayscale hand part filters

(d) depth hand part filters

Fig. 8: Visualization of *conv1* layer for different input channels. It can be seen that the hand filters are smoother than the body part filters because hand input are smoother than the body part input.

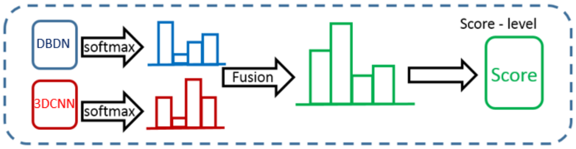


Fig. 9: Illustration of descriptor fusion.

noisy tokens is to discard the token path log probability small than certain threshold. However, because the metric of this challenge: *Jaccard index* strongly penalizes false negatives, experiments show that it's better to have more false positives than to miss true positives. Effective ways to detect false positives should be an interesting aspect of future works.

3.4 Score Fusion

To fuse the dual model prediction, the strategy shown as Fig 9 is adopted. The complementary properties of both modules can be seen from the Viterbi path decoding plot in Fig 12 :

$$\mathbf{S} = \alpha * \mathbf{S^1} + (1-\alpha) * \mathbf{S^2} \tag{12}$$

where $\mathbf{S^1}$ and $\mathbf{S^2}$ are the score probability matrix as in Algo. 2, corresponding to the skeletal input and depth & RGB input, and α is the coefficient controls the score balance obtained by cross validation. Interestingly, the best performing α is close to 0.5. Thus indicating that both approaches perform comparably.

The individual module result and the fusion result are shown in Tab. 1. Note that the skeleton module generally performs better than the depth module, one reason could be that the skeleton joints learnt from [23] lie in success of utilizing huge and highly varied training data: from both realistic and synthetic depth images, a total number of 1 million images were used to train the deep randomized decision forest classifier in order to avoid overfitting. Hence skeleton data are more robust.

Module \ Evaluation Set	Validation	Test
Skeleton–DBDN	0.78266	0.77920
Depth & RGB –3DCNN	0.75163	0.71678
Score Fusion	0.81744	0.80910

TABLE 1: Comparison of results in terms of Jaccard index between different network structures and various modules.

		Validation	Test
skeleton	ACC	0.8633	0.8360
	UnRate	0.0230	0.0412
depth & RGB	ACC	0.7871	0.7581
	UnRate	0.1612	0.1976
score fusion	ACC	0.8791	0.8642
	UnRate	0.0302	0.0485

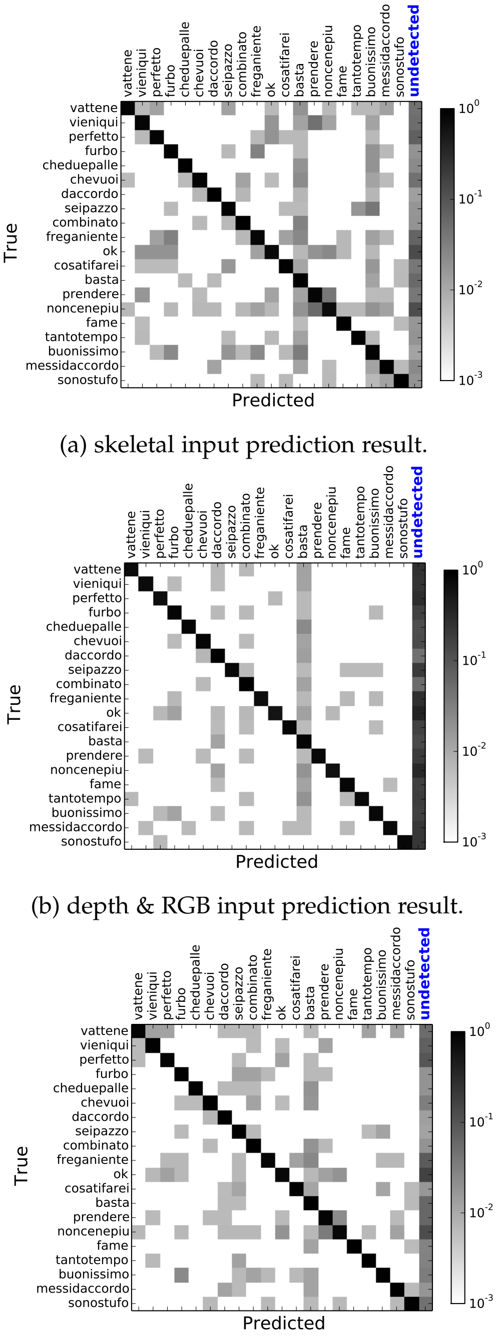
TABLE 2: Gesture classification accuracy and undetected rate: if prediction overlaps with the ground truth more than 50%, it's counted as true positive. Acc: classification accuracy, UnRate: undetected Rate.

From the frame based prediction, we also evaluate the gesture token classification rate using the commonly-used PASCAL overlap criterion: if the gesture is predicted correctly with more than 50% overlap with the ground truth label, then the prediction is counted as a true positive. The results of two individual module and the score fused module are shown in Tab. 2 and the confusion matrices are shown in Fig. 10. While many of the gestures in this dataset could be mainly differentiated by examining the positions and motions of large joints such as the elbows and wrists, some gestures differ primarily in hand pose, *e.g.*, Fig. 11. From the confusion matrices in Fig. 10 we can observe the complimentary information between the skeleton input and the depth & RGB input.

4 Related Works

This paper served as the progressive works of [37] and [38]. Ji *et al.* [19] proposed using 3D Convolutional Neural Network for automated recognition of human actions in surveillance videos. Their model extracts features from both the spatial and the temporal dimensions by performing 3D convolutions, thereby capturing the motion information encoded in multiple adjacent frames. To further boost the performance, they proposed regularizing the outputs with high-level features and combining the predictions of a variety of difference models.

In the same challenge, [39] present a multi-scale and multi-modal deep networks for gesture detection and localization. Key to their technique is a training strategy that exploits i) careful initialization of initialization of individual modalities and ii) gradual fusion of modalities from strongest to weakest cross-modality structure. One major difference between our proposed method and theirs is the treatment of time factor. In [39], the proposed networks

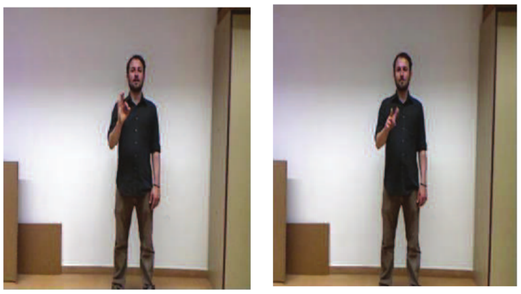


(a) skeletal input prediction result.

(b) depth & RGB input prediction result.

(c) score fusion prediction result.

Fig. 10: Confusion Matrix of skeletal input, depth & RGB input and score fusion result. Generally skeleton module has higher classification rate. Some gestures, *e.g.* "OK" and ”Non ce ne piu" differ primarily in hand poses. Hence they are easier to be differentiated using depth & RGB module (middle confusion matrix) than the skeleton module (top confusion matrix).



(a) G11 "OK" (b) G15 "Non ce ne piu"

Fig. 11: Examples of gestures that differ primarily in hand pose but not the arm motions.

requires fixed length inputs and in order to cope with various action speed, a multiscale network is required which potentially increases the time for training and testing.

Some of the top winning methods in this challenges requires a set of complicated handcrafted features for either skeletal input, depth & RGB input, or both. *E.g.*, [39] formulated a pose descriptor, consiting of 7 logical subsets. [40] proposed four types of features for skeleton features: normalized joint positions; joint quaternion angles; Euclidean distances between specific joints; and directed distances between pairs of joints, based on the features proposed by Yao *et al.* [41] and a histogram of oriented gradients (HOG) descriptor for hand features. In [42], the state-of-the-art dense trajectory handcrafted features are adopted for the RGB module. Multiple networks averaging works better than an single individual network and it can be seen from the experiments in [37] that larger net will generally perform better than smaller net, averaging multi-column nets almost will certainly further improve the performance [17].

5 Experiments and Analysis

5.1 Chalearn LAP Dataset & Evaluation Metrics

The dataset[1] used in this work contains 20 different gestures. These gestures are selected from Italian cultural sign gestures. For each of the 20 gestures, there are 700 examples available for training and 240 sequences are used for testing. The testing sequences however are not segmented a priori and the actions must be detected within a continuous data stream. This dataset is on "multiple instance, user independent learning and continuous gesture spotting" [25] of gestures. In the 3 track, there are more than 14,000 gestures.

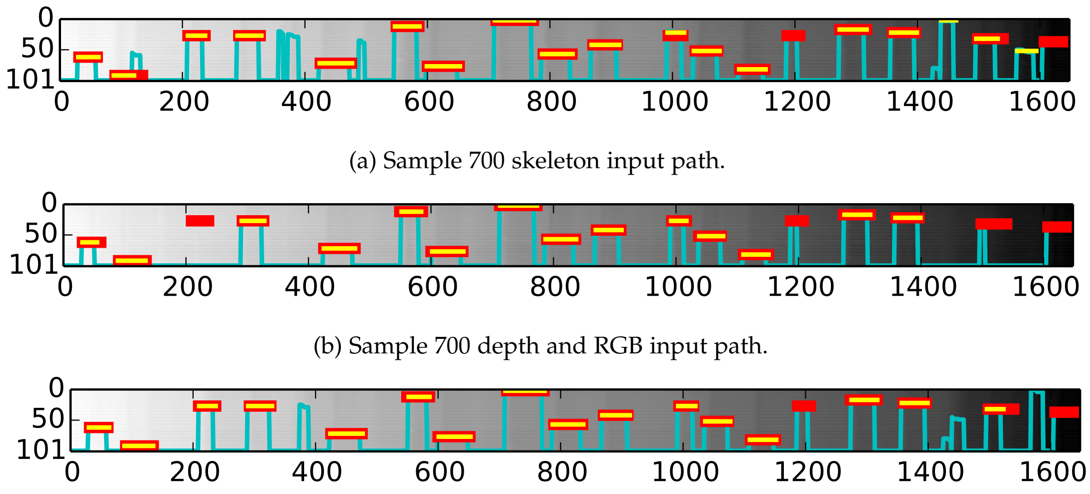
For input sequences, there are three modalities, *i.e.*, skeleton, RGB and depth (with user segmentation) provided. In the following experiments, the first 650 sample sequences are used for training, 50 for validation and the rest 240 for testing where each sequence contains around 20 gestures with some noisy non-meaningful vocabulary tokens.

The evaluation of this dataset is performed using the *Jaccard* index, which computed the overlap between the ground truth and the predictions on a frame by frame basis:

$$J(A,B) = \frac{A \bigcap B}{A \bigcup B} \tag{13}$$

where A is the ground truth gesture label and B is the predicted gesture label.

[1] http://gesture.chalearn.org/homewebsourcereferrals



(a) Sample 700 skeleton input path.

(b) Sample 700 depth and RGB input path.

(c) Sample 700 combined input path.

Fig. 12: Viterbi decoding of two modules and their fusion result of sample sequence 700. Top to bottom: skeleton, depth & RGB, score fusion with x-axis representing the time and y-axis representing the hidden states of all the classes with the ergodic state at the bottom. Red lines are the ground truth label, cyan lines are the viterbi shortest path and yellow lines are the predicted label. There are some complementary information of the two modules and generally skeletal module outperforms the depth module. The fusion of the two could exploit the uncertainty, *e.g.*, around frame 200 skeleton can help with the false negative prediction given by CNN module and around frame 1450, CNN module can help suppress the false positive prediction given by skeleton module.

Evaluation Set Module	skeleton	Depth & RGB	Fusion
[39] deep learning (step 2)	0.6938	0.7862	**0.8500**
[40] 4 set skeletal & HOG	0.7420	-	0.8220
[43] hand crafted	**0.7948**	-	0.8268
[42] dense trajectory	-	**0.7919**	-
[38] cnn	-	0.7888	-
[37] deep learning	0.7468	0.6371	0.8045
DDNN (this work)	0.7792	0.7168	0.8091

TABLE 3: Comparison of results in terms of Jaccard index between different network structures and various modules.

5.2 Computational complexity

Though learning the Deep Neural Networks using stochastic gradient descent is computationally intensive, once the model finishes training, with a low inference cost, our framework can perform real-time video sequence labeling. Specifically, a single feed forward neural network incurs trivial computational time ($\mathcal{O}(T)$) and is fast because it requires only matrix products and convolution operations. The complexity of Viterbi algorithm is $\mathcal{O}(T * |S|^2)$ with number of frames T and state number $|S|$.

6 Conclusion and Discussion

Hand-engineered, task-specific features are often less adaptive and time-consuming to design. This difficulty is more pronounced with multimodal data as the features have to relate multiple data sources. In this paper, we presented a novel Deep Dynamic Neural Networks(DDNN) framework that utilizes Deep Belief Networks and 3D Convolutional Neural Networks for learning contextual frame-level representations and modeling emission probabilities for Markov Field. The heterogeneous inputs from skeletal joints, RGB and depth images require different feature learning methods and the late fusion scheme is adopted at the score level. The experimental results on bi-modal time series data show that the multimodal DDNN framework can learn a good model of the joint space of multiple sensory inputs, and is consistently as good as/better than the unimodal input, opening the door for exploring the complementary representation among multimodal inputs. It also suggests that learning features directly from data is a very important research direction and with more and more data and flops-free computational power, the learning-based methods are not only more generalizable to many domains, but also are powerful in combining with other well-studied probabilistic graphical models for modeling and reasoning dynamic sequences. Future works include learning the share representation amongst the heterogeneous inputs at the penultimate layer and backpropagating the gradient in the share space in a unified representation.

Appendix A
Details of the Code

The python project for can be found at:
`https://github.com/stevenwudi/chalearn2014_wudi_lio`

Acknowledgments

The authors would like to thank Sander Dieleman for his guidance in building, training and initializing convolutional neural networks.

References

[1] L. Liu, L. Shao, F. Zheng, and X. Li, “Realistic action recognition via sparsely-constructed gaussian processes,” *Pattern Recognition, doi: 10.1016/j.patcog.2014.07.006.*, 2014.

[2] L. Shao, X. Zhen, and X. Li, "Spatio-temporal laplacian pyramid coding for action recognition," *IEEE Transactions on Cybernetics, vol. 44, no. 6, pp. 817-827*, 2014.

[3] D. Wu and L. Shao, "Silhouette analysis-based action recognition via exploiting human poses," *IEEE Transactions on Circuits and Systems for Video Technology, vol. 23, no. 2, pp. 236-243*, 2013.

[4] I. Laptev, "On space-time interest points," *International Journal of Computer Vision*, 2005.

[5] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *Visual Surveillance and Performance Evaluation of Tracking and Surveillance*. IEEE, 2005.

[6] G. Willems, T. Tuytelaars, and L. V. Gool, "An efficient dense and scale-invariant spatio-temporal interest point detector," in *European Conference on Computer Vision*. Springer, 2008.

[7] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in *International Conference on Multimedia*. ACM, 2007.

[8] A. Klaser, M. Marszalek, and C. Schmid, "A Spatio-Temporal Descriptor Based on 3D-Gradients," in *British Machine Vision Conference*, 2008.

[9] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," *International Journal of Computer Vision*, 2013.

[10] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, C. Schmid *et al.*, "Evaluation of local spatio-temporal features for action recognition," in *British Machine Vision Conference*, 2009.

[11] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler, "Convolutional learning of spatio-temporal features," in *European Conference on Computer Vision*. Springer, 2010.

[12] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng, "Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[13] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt, "Spatio-temporal convolutional sparse auto-encoder for sequence classification," in *British Machine Vision Conference*, 2012.

[14] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, 2006.

[15] J. Schmidhuber, "Deep learning in neural networks: An overview," *arXiv preprint arXiv:1404.7828*, 2014.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Neural Information Processing Systems*, 2012.

[17] D. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[18] M. Y. Shuiwang Ji, Wei Xu and K. Yu, "3d convolutional neural networks for human action recognition," in *International Conference on Machine Learning*. IEEE, 2010.

[19] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2013.

[20] A. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *Audio, Speech, and Language Processing, IEEE Transactions on*, 2012.

[21] D. Wu and L. Shao, "Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[22] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates *et al.*, "Deepspeech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[23] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[24] J. Han, L. Shao, and J. Shotton, "Enhanced computer vision with microsoft kinect sensor: A review," *IEEE Transactions on Cybernetics, vol. 43, no. 5, pp. 1317-1333*, 2013.

[25] S. Escalera, J. Gonzlez, X. Bar, M. Reyes, O. Lops, I. Guyon, V. Athitsos, and H. J. Escalante, "Multi-modal gesture recognition challenge 2013: Dataset and results." in *ACM ChaLearn Multi-Modal Gesture Recognition Grand Challenge and Workshop*, 2013. [Online]. Available: http://gesture.chalearn.org/

[26] S. Fothergill, H. M. Mentis, P. Kohli, and S. Nowozin, "Instructing people for training gestural interactive systems," in *ACM Computer Human Interaction*, 2012.

[27] I. Guyon, V. Athitsos, P. Jangyodsuk, B. Hamner, and H. J. Escalante, "Chalearn gesture challenge: Design and first results," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2012.

[28] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Mining actionlet ensemble for action recognition with depth cameras," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[29] A. Lehrmann, P. Gehler, and S. Nowozin, "A non-parametric bayesian network prior of human pose," in *International Conference on Computer Vision*, 2013.

[30] S. Nowozin and J. Shotton, "Action points: A representation for low-latency online human action recognition," Tech. Rep., 2012.

[31] C. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[32] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[33] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[34] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 2146–2153.

[35] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[36] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 2013, pp. 1139–1147.

[37] D. Wu and L. Shao, "Deep dynamic neural networks for gesture segmentation and recognition," *European Conference on Computer Vision and Pattern Recognition Workshops*, 2014.

[38] L. Pigou, S. Dieleman, P.-J. Kindermans, and B. Schrauwen, "Sign language recognition using convolutional neural networks," *European Conference on Computer Vision and Pattern Recognition Workshops*, 2014.

[39] N. Neverova, C. Wolf, G. Taylor, and F. Nebout, "Multi-scale deep learning for gesture detection and localization," in *European Conference on Computer Vision and Pattern Recognition Workshops*, 2014.

[40] C. Monnier, S. German, and A. Ost, "A multi-scale boosted detector for efficient and robust gesture recognition," in *European Conference on Computer Vision and Pattern Recognition Workshops*, 2014.

[41] A. Yao, J. Gall, G. Fanelli, and L. J. Van Gool, "Does human action recognition benefit from pose estimation?." in *BMVC*, 2011.

[42] X. Peng, L. Wang, and Z. Cai, "Action and gesture temporal spotting with super vector representation," in *European Conference on Computer Vision and Pattern Recognition Workshops*, 2014.

[43] J. Y. Chang, "Nonparametric gesture labeling from multi-modal data," in *European Conference on Computer Vision and Pattern Recognition Workshops*, 2014.