Factored Temporal Sigmoid Belief Networks for Sequence Learning: Supplementary Material

Jiaming Song[†] JIAMING.TSONG@GMAIL.COM
Zhe Gan[‡] ZHE.GAN@DUKE.EDU
Lawrence Carin[‡] LCARIN@DUKE.EDU
[†]Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, China
[‡]Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708, USA

A. Variants of Conditional TSBNs

In the main paper, we considered modeling real-valued sequence data. Other forms of data, such as binary and count data, can also be modeled by slight modification of the model.

Modeling binary data Our models can be readily extended to model binary sequence data, by substituting $p(\boldsymbol{v}_t|\boldsymbol{v}_{t-1}, \boldsymbol{h}_t, \boldsymbol{y}_t) = \text{Ber}(\boldsymbol{v}_t; \sigma(\tilde{\boldsymbol{v}}_t))$, where

$$\tilde{\boldsymbol{v}}_t = \mathbf{W}_2^{(y)}\boldsymbol{h}_t + \mathbf{W}_4^{(y)}\boldsymbol{v}_{t-1} + \boldsymbol{c}^{(y)} \tag{1}$$

$\sigma(x) = 1/(1+\exp(-x))$, and $\text{Ber}(x;p)$ denotes the Bernoulli distribution with parameter p.

Modeling count data We also introduce an approach for modeling time-series data with count observations, $p(\boldsymbol{v}_t|\boldsymbol{v}_{t-1}, \boldsymbol{h}_t, \boldsymbol{y}_t) = \prod_{m=1}^{M} s_{mt}^{v_{mt}}$, where

$$s_{mt} = \frac{\exp(\boldsymbol{h}_t^\top \boldsymbol{w}_{2m}^{(y)} + \boldsymbol{v}_{t-1}^\top \boldsymbol{w}_{4m}^{(y)} + c_m^{(y)})}{\sum_{m'=1}^{M} \exp(\boldsymbol{h}_t^\top \boldsymbol{w}_{2m'}^{(y)} + \boldsymbol{v}_{t-1}^\top \boldsymbol{w}_{4m'}^{(y)} + c_{m'}^{(y)})} \tag{2}$$

B. Inference Details

B.1. Outline of NVIL Algorithm

The outline of the NVIL Algorithm for computing gradients are shown in Algorithm 1. $C_{\boldsymbol{\lambda}}(\boldsymbol{v}_t)$ represents the sum of data-dependent baseline and data-independent baseline.

B.2. Derivatives for Conditional TSBNs

For the Conditional TSBNs, we have:

$$p(h_{jt} = 1|\boldsymbol{h}_{t-1}, \boldsymbol{v}_{t-1}, \boldsymbol{y}_t) = \sigma(\tilde{h}_{jt}) \tag{3}$$

$$p(\boldsymbol{v}_t|\boldsymbol{h}_t, \boldsymbol{v}_{t-1}, \boldsymbol{y}_t) = \mathcal{N}(\boldsymbol{\mu}_t, \text{diag}(\boldsymbol{\sigma}_t^2)) \tag{4}$$

$$\tilde{\boldsymbol{h}}_t = \mathbf{W}_1^{(y)}\boldsymbol{h}_{t-1} + \mathbf{W}_3^{(y)}\boldsymbol{v}_{t-1} + \boldsymbol{b}^{(y)} \tag{5}$$

$$\boldsymbol{\mu}_t = \mathbf{W}_2^{(y)}\boldsymbol{h}_t + \mathbf{W}_4^{(y)}\boldsymbol{v}_{t-1} + \boldsymbol{c}^{(y)} \tag{6}$$

$$\log \boldsymbol{\sigma}_t^2 = \mathbf{W}_2'^{(y)}\boldsymbol{h}_t + \mathbf{W}_4'^{(y)}\boldsymbol{v}_{t-1} + \boldsymbol{c}'^{(y)} \tag{7}$$

Algorithm 1 Calculate gradient estimates for model parameters and recognition parameters.

$\Delta\boldsymbol{\theta} \leftarrow \mathbf{0}$, $\Delta\boldsymbol{\phi} \leftarrow \mathbf{0}$, $\Delta\boldsymbol{\lambda} \leftarrow \mathbf{0}$
$\kappa \leftarrow 0$, $\tau \leftarrow 0$
$\mathcal{L} \leftarrow \mathbf{0}$
for $t \leftarrow 1$ **to** T **do**
  $\boldsymbol{h}_t \sim q_\phi(\boldsymbol{h}_t|\boldsymbol{v}_t)$
  $l_t \leftarrow \log p_\theta(\boldsymbol{v}_t, \boldsymbol{h}_t) - \log q_\theta(\boldsymbol{h}_t|\boldsymbol{v}_t)$
  $\mathcal{L} \leftarrow \mathcal{L} + l_t$
  $l_t \leftarrow l_t - C_{\boldsymbol{\lambda}}(\boldsymbol{v}_t)$
end for
$\kappa_b \leftarrow \text{mean}(l_1, \ldots, l_T)$
$\tau_b \leftarrow \text{var}(l_1, \ldots, l_T)$
$\kappa \leftarrow \rho\kappa + (1-\rho)\kappa_b$
$\tau \leftarrow \rho\tau + (1-\rho)\tau_b$
for $t \leftarrow 1$ **to** T **do**
  $l_t \leftarrow \frac{l_t - \kappa}{\max(1, \sqrt{\tau})}$
  $\Delta\boldsymbol{\theta} \leftarrow \Delta\boldsymbol{\theta} + \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{v}_t, \boldsymbol{h}_t)$
  $\Delta\boldsymbol{\phi} \leftarrow \Delta\boldsymbol{\phi} + l_t \nabla_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{h}_t|\boldsymbol{v}_t)$
  $\Delta\boldsymbol{\lambda} \leftarrow \Delta\boldsymbol{\lambda} + l_t \nabla_{\boldsymbol{\lambda}} C_{\boldsymbol{\lambda}}(\boldsymbol{v}_t)$
end for

The recognition model is expressed as:

$$q(h_{jt} = 1|\boldsymbol{h}_{t-1}, \boldsymbol{v}_t, \boldsymbol{v}_{t-1}, \boldsymbol{y}_t) = \sigma(\hat{h}_{jt}) \tag{8}$$

$$\hat{\boldsymbol{h}}_t = \mathbf{U}_1^{(y)}\boldsymbol{h}_{t-1} + \mathbf{U}_2^{(y)}\boldsymbol{v}_t + \mathbf{U}_3^{(y)}\boldsymbol{v}_{t-1} + \boldsymbol{d}^{(y)} \tag{9}$$

In order to implement the NVIL algorithm, we need to calculate the lower bound and also the gradients. Specifically, the lower bound can be expressed as $\mathcal{L} = \sum_{t=1}^{T} \mathbb{E}_{q_\phi(\mathbf{H}|\mathbf{V})}[l_t]$, where

$$l_t = \sum_{j=1}^{J} (\tilde{h}_{jt} h_{jt} - \log(1 + \exp(\tilde{h}_{jt}))) \tag{10}$$

$$+ \sum_{m=1}^{M} (\frac{1}{2}\log 2\pi + \log \sigma_{mt} + \frac{(v_{mt} - \mu_{mt})^2}{2\sigma_{mt}^2}) \tag{11}$$

$$+\sum_{j=1}^{M}(\hat{h}_{jt}h_{jt}-\log(1+\exp(\hat{h}_{jt}))) \tag{12}$$

The gradients for model parameters θ are

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \hat{\mathbf{W}}_{1jj's}}=(h_{jt}-\sigma(\tilde{h}_{jt}))\cdot h_{j'(t-1)}\cdot y_{st} \tag{13}$$

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \hat{\mathbf{W}}_{2mjs}}=\frac{v_{mt}-\mu_{mt}}{\sigma_{mt}^2}\cdot h_{jt}\cdot y_{st} \tag{14}$$

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \hat{\mathbf{W}}'_{2mjs}}=(\frac{(v_{mt}-\mu_{mt})^2}{\sigma_{mt}^2}-1)\cdot h_{jt}\cdot y_{st} \tag{15}$$

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \hat{\mathbf{W}}_{3jms}}=(h_{jt}-\sigma(\tilde{h}_{jt}))\cdot v_{m(t-1)}\cdot y_{st} \tag{16}$$

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \hat{\mathbf{W}}_{4mm's}}=\frac{v_{mt}-\mu_{mt}}{\sigma_{mt}^2}\cdot v_{m'(t-1)}\cdot y_{st} \tag{17}$$

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \hat{\mathbf{W}}'_{4mm's}}=(\frac{(v_{mt}-\mu_{mt})^2}{\sigma_{mt}^2}-1)\cdot v_{m'(t-1)}\cdot y_{st} \tag{18}$$

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \mathbf{B}_{js}}=(h_{jt}-\sigma(\tilde{h}_{jt}))\cdot y_{st} \tag{19}$$

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \mathbf{C}_{ms}}=(\frac{v_{mt}-\mu_{mt}}{\sigma_{mt}^2}\cdot y_{st})\cdot y_{st} \tag{20}$$

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \mathbf{C}'_{ms}}=(\frac{(v_{mt}-\mu_{mt})^2}{\sigma_{mt}^2}-1)\cdot y_{st} \tag{21}$$

The gradients for recognition parameters ϕ are

$$\frac{\partial \log q_\phi(\boldsymbol{h}_t|\boldsymbol{v}_t)}{\partial \hat{\mathbf{U}}_{1jj's}}=(h_{jt}-\sigma(\hat{h}_{jt}))\cdot h_{j'(t-1)}\cdot y_{st} \tag{22}$$

$$\frac{\partial \log q_\phi(\boldsymbol{h}_t|\boldsymbol{v}_t)}{\partial \hat{\mathbf{U}}_{2jms}}=(h_{jt}-\sigma(\hat{h}_{jt}))\cdot h_{mt}\cdot y_{st} \tag{23}$$

$$\frac{\partial \log q_\phi(\boldsymbol{h}_t|\boldsymbol{v}_t)}{\partial \hat{\mathbf{U}}_{3jms}}=(h_{jt}-\sigma(\hat{h}_{jt}))\cdot v_{m(t-1)}\cdot y_{st} \tag{24}$$

$$\frac{\partial \log q_\phi(\boldsymbol{h}_t|\boldsymbol{v}_t)}{\partial \hat{\mathbf{D}}_{js}}=(h_{jt}-\sigma(\hat{h}_{jt}))\cdot y_{st} \tag{25}$$

B.3. Derivatives for the Factored Model

The factored model substitutes the weight tensors $\hat{\mathbf{W}}$ with three matrices $\mathbf{W}_a$, $\mathbf{W}_b$, and $\mathbf{W}_c$, such that

$$\mathbf{W}^{(y)}=\mathbf{W}_a\cdot\text{diag}(\mathbf{W}_b\boldsymbol{y}_t)\cdot\mathbf{W}_c \tag{26}$$

We notice that for a particular $\mathbf{W}^{(y)}$, the objective function for the t-th time step can be generalized as $\mathcal{L}'=f(\mathbf{W}^{(y)}\boldsymbol{\eta}+\boldsymbol{\chi})+\rho$, where the elements of $\frac{\partial\boldsymbol{\eta}}{\mathbf{W}^{(y)}}$, $\frac{\partial\boldsymbol{\chi}}{\mathbf{W}^{(y)}}$ and $\frac{\partial\rho}{\mathbf{W}^{(y)}}$ are zero. Assuming $\boldsymbol{\xi}=f'(\mathbf{W}^{(y)}\boldsymbol{\eta}+\boldsymbol{\chi})$, we have the following gradients:

$$\frac{\partial\mathcal{L}'}{\partial\mathbf{W}_a}=\boldsymbol{\xi}\cdot(\text{diag}(\mathbf{W}_b\boldsymbol{y}_t)\cdot\mathbf{W}_c\boldsymbol{\eta})^\top \tag{27}$$

$$\frac{\partial\mathcal{L}'}{\partial\mathbf{W}_b}=((\mathbf{W}_a^\top\boldsymbol{\xi})\odot(\mathbf{W}_c\boldsymbol{\eta}))\cdot\boldsymbol{y}_t^\top \tag{28}$$

$$\frac{\partial\mathcal{L}'}{\partial\mathbf{W}_c}=\text{diag}(\mathbf{W}_b\boldsymbol{y}_t)\cdot\mathbf{W}_a\cdot\boldsymbol{\xi}\boldsymbol{\eta}^\top \tag{29}$$

where $\mathbf{A}\odot\mathbf{B}$ denotes the element-wise product between matrices $\mathbf{A}$ and $\mathbf{B}$ with the same dimensions.

For FCTSBN, the gradients for bias parameters in 19, 20, 21 and 25 remains the same, while gradients for the factored weight parameters can be calculated using 27 - 29. For $\mathbf{W}_{1a}$, $\mathbf{W}_{1b}$ and $\mathbf{W}_{1c}$, we have $\boldsymbol{\xi}=\boldsymbol{h}_t-\sigma(\tilde{\boldsymbol{h}}_t)$ and $\boldsymbol{\eta}=\boldsymbol{h}_{t-1}$. Hence the gradients for these parameters are:

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \mathbf{W}_{1a}}=(\boldsymbol{h}_t-\sigma(\tilde{\boldsymbol{h}}_t))\cdot(\text{diag}(\mathbf{W}_{1b}\boldsymbol{y}_t)\cdot\mathbf{W}_{1c}\boldsymbol{h}_{t-1})^\top \tag{30}$$

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \mathbf{W}_{1b}}=((\mathbf{W}_{1a}^\top(\boldsymbol{h}_t-\sigma(\tilde{\boldsymbol{h}}_t)))\odot(\mathbf{W}_{1c}\boldsymbol{h}_{t-1}))\cdot\boldsymbol{y}_t^\top \tag{31}$$

$$\frac{\partial \log p_\theta(\boldsymbol{v}_t,\boldsymbol{h}_t)}{\partial \mathbf{W}_{1c}}=\text{diag}(\mathbf{W}_{1b}\boldsymbol{y}_t)\cdot\mathbf{W}_{1a}\cdot(\boldsymbol{h}_t-\sigma(\tilde{\boldsymbol{h}}_t))\boldsymbol{h}_{t-1}^\top \tag{32}$$

Gradients of other factored parameters can be derived in analogy.

B.4. Gradients for Deep Models

Suppose we have a two-layered deep CTSBN with the joint probability distribution $p_\theta(\mathbf{V},\mathbf{H},\mathbf{Z})$. $\mathbf{Z}=[\boldsymbol{z}_1,\cdot,\boldsymbol{z}_T]$, where $\boldsymbol{z}_t\in\{0,1\}^K$ is the latent variables on the second layer at time t. The probability of $\boldsymbol{z}_t$ and $\boldsymbol{h}_t$ are characterized by

$$p(z_{kt}=1|\boldsymbol{h}_{t-1},\boldsymbol{z}_{t-1},\boldsymbol{y}_t)=\sigma(\tilde{z}_{et}) \tag{33}$$

$$p(h_{jt}=1|\boldsymbol{h}_{t-1},\boldsymbol{z}_t,\boldsymbol{v}_{t-1},\boldsymbol{y}_t)=\sigma(\tilde{h}_{jt}) \tag{34}$$

$$\tilde{\boldsymbol{z}}_v=\mathbf{W}_6^{(y)}\boldsymbol{h}_{t-1}+\mathbf{W}_7^{(y)}\boldsymbol{z}_{t-1}+\boldsymbol{a}^{(y)} \tag{35}$$

$$\tilde{\boldsymbol{h}}_t=\mathbf{W}_1^{(y)}\boldsymbol{h}_{t-1}+\mathbf{W}_3^{(y)}\boldsymbol{v}_{t-1}+\mathbf{W}_5^{(y)}\boldsymbol{z}_t+\boldsymbol{b}^{(y)} \tag{36}$$

The recognition model for $\boldsymbol{z}_t$ is expressed as:

$$q(z_{kt}=1|\boldsymbol{z}_{t-1},\boldsymbol{h}_t,\boldsymbol{h}_{t-1},\boldsymbol{y}_t)=\sigma(\hat{z}_{kt}) \tag{37}$$

$$\hat{\boldsymbol{z}}_t=\mathbf{U}_4^{(y)}\boldsymbol{z}_{t-1}+\mathbf{U}_5^{(y)}\boldsymbol{h}_t+\mathbf{U}_6^{(y)}\boldsymbol{h}_{t-1}+\boldsymbol{e}^{(y)} \tag{38}$$

The lower bound takes the form $\mathcal{L}=\sum_{t=1}^T\mathbb{E}_{q_\phi(\mathbf{Z},\mathbf{H}|\mathbf{V})}[l_t]$, where

$$l_t=\sum_{j=1}^{J}(\tilde{h}_{jt}h_{jt}-\log(1+\exp(\tilde{h}_{jt}))) \tag{39}$$

$$+\sum_{m=1}^{M}(\frac{1}{2}\log 2\pi+\log\sigma_{mt}+\frac{(v_{mt}-\mu_{mt})^2}{2\sigma_{mt}^2}) \tag{40}$$

$$+\sum_{j=1}^{M}(\hat{h}_{jt}h_{jt}-\log(1+\exp(\hat{h}_{jt}))) \quad (41)$$

$$+\sum_{k=1}^{K}(\tilde{z}_{kt}z_{kt}-\log(1+\exp(\tilde{z}_{kt}))) \quad (42)$$

$$+\sum_{k=1}^{K}(\hat{z}_{kt}z_{kt}-\log(1+\exp(\hat{z}_{kt}))) \quad (43)$$

Compared with the one-layer model, the two-layer model adds two terms concerning $\boldsymbol{z}_t$, whose form is similar to that of $\boldsymbol{h}_t$. Therefore, gradients for additional parameters ($\hat{\mathbf{W}}_5$, $\hat{\mathbf{W}}_6$, $\hat{\mathbf{W}}_7$, $\hat{\mathbf{U}}_4$, $\hat{\mathbf{U}}_5$, $\hat{\mathbf{U}}_6$, $\mathbf{A}$ and $\mathbf{E}$) can be readily calculated using Equations 13, 22, 19. If the model parameters are factored, gradients can be calculated using 27-29.

B.5. Semi-supervised FCTSBN

The lower bound for semi-supervised learning $\mathcal{L}_s$ can be described as $\mathcal{L}_s = \mathcal{L}_l + \mathcal{L}_u$, where $\mathcal{L}_l$ and $\mathcal{L}_u$ are the lower bounds for labeled data and unlabeled data respectively:

$$\mathcal{L}_l = \mathcal{L} + \alpha \cdot \mathbb{E}_{\tilde{p}_l(\mathbf{V},\mathbf{Y})}[\log q_{\boldsymbol{\theta}}(\mathbf{Y}|\mathbf{V})] \quad (44)$$

$$\mathcal{L}_u = \mathcal{J}(q_{\boldsymbol{\phi}}(\mathbf{H},\mathbf{Y}|\mathbf{V}), p_{\boldsymbol{\theta}}(\mathbf{H},\mathbf{V},\mathbf{Y})) \quad (45)$$

For labeled data, $\mathcal{L}_l$ adds a classification loss to the generative model lower bound, where the gradient can be readily calculated from Algorithm 1 plus a term $\alpha\nabla_{\boldsymbol{\theta}}\mathbb{E}_{\tilde{p}_l(\mathbf{V},\mathbf{Y})}[\log q_{\boldsymbol{\theta}}(\mathbf{Y}|\mathbf{V})]$, which can also be approximated using Monte-Carlo integration. For unlabeled data, $\mathcal{L}_u$ requires calculating the expectation with respect to $q_{\boldsymbol{\phi}}(\mathbf{H},\mathbf{Y}|\mathbf{V}) = q_{\boldsymbol{\phi}}(\mathbf{H}|\mathbf{V},\mathbf{Y})q_{\boldsymbol{\phi}}(\mathbf{Y}|\mathbf{V})$. Hence, to optimize $\mathcal{L}_u$, we can sample $\mathbf{Y}$ from $\mathbf{V}$ first, and then apply Algorithm 1 to obtain the approximated gradients. Outline for optimizing $\mathcal{L}_s$ is shown in Algorithm 2.

Algorithm 2 Optimizing the semi-supervised objective $\mathcal{L}_s$.

```
Initialize θ, φ
while not converged do
  if sample labeled data then
    (v, y) ~ p̃_l(V, Y)
  else
    v ~ p̃_u(V)
    y ~ q_φ(y|v)
  end if
  Calculate ∇_θ L_s and ∇_φ L_s
  θ ← θ + ∇_θ L_s
  φ ← φ + ∇_φ L_s
end while
```

B.6. Computational Complexity

Although side information is included, the computational complexity of learning and inference for CTSBN and FCTSBN are comparable to that of TSBN. Suppose that $\boldsymbol{y}_t$ are one-hot encoded vectors, so gradients of some parameters are zero and does not cost computation. For CTSBN, the complexity for calculating the gradient would be $O\big((J+M)^2mn\big)$, where m is the size of the mini batch, and n is the order. For FCTSBN, the complexity would be $O\big(F(J+M)mn\big)$.

Empirically, training for even the largest model in our experiments takes around 10 hours on unoptimized MATLAB using a laptop, whereas generating thousands of samples can be achieved within seconds.

Style	FCTSBN	dFCTSBN	FCRBM
Sexy	0.4971	0.2401	0.4926
Strong	0.2899	0.2415	0.3385
Cat	0.1858	0.1732	0.3475
Dinosaur	0.6299	0.4182	0.3387
Drunk	0.6227	0.6184	0.5005
Gangly	0.5553	0.3777	0.5474
Chicken	0.7798	0.6909	0.3519
Graceful	0.7184	0.4232	0.3544
Normal	0.3043	0.2330	0.2713
Old man	0.2483	0.1831	0.8125
Average	0.4832	0.3599	0.4355

Table 1: Average prediction error for mocap10

C. Extended Experiment Results

C.1. Generated Videos

Along with this supplementary article including more details for our model, we present a number of videos to demonstrate the generative capacities of our models.

mocap2 We present synthesized sequences by the semi-supervised Hidden Markov FCTSBN trained with labeled and unlabeled data, namely **walk.mp4**, **run.mp4**, **walk-run.mp4** and **run-walk.mp4**.

The videos denotes sequences of (*i*) walking; (*ii*) running; (*iii*) transition from walking to running; and (*iv*) transition from running to walking.

mocap10 Sequences produced by the Hidden Markov FCTSBN over the mocap10 dataset are presented, including 9 styles and some style transitions and combinations.

C.2. Detailed Results on mocap10 Prediction

Average prediction error for each style can be found in Table 1. For the FCTSBN and deep FCTSBN, each hidden layer has 100 hidden variables and 50 factors, whereas the FCRBM contains 600 hidden variables, 200 features and

% of labeled data	10-KNN	Softmax-0.001	Softmax-1	TSVM	FCTSBN	dFCTSBN
0.25	73.16	72.07 ± 0.52	71.44 ± 0.55	76.53	78.33 ± 1.75	79.95 ± 2.04
0.3125	78.00	78.11 ± 0.21	78.07 ± 1.34	79.47	81.11 ± 1.22	82.94 ± 1.08
0.375	81.37	82.31 ± 0.58	80.53 ± 0.73	81.47	83.74 ± 1.03	86.10 ± 1.53
0.4375	83.68	83.26 ± 0.84	80.91 ± 2.06	82.22	84.37 ± 2.96	87.20 ± 1.28
0.5	84.53	83.81 ± 0.86	81.26 ± 1.42	83.37	85.17 ± 1.84	87.70 ± 2.16
0.5625	86.00	85.11 ± 2.90	81.28 ± 1.64	84.31	86.84 ± 1.18	88.16 ± 1.25
0.625	85.47	85.78 ± 0.95	84.94 ± 0.74	86.94	88.17 ± 1.66	89.40 ± 1.55
0.6875	86.00	86.03 ± 1.49	83.95 ± 1.26	86.63	88.63 ± 1.45	89.40 ± 1.96
0.75	84.74	85.98 ± 1.51	85.07 ± 0.88	87.26	88.73 ± 1.68	89.57 ± 1.89

Table 2: Test accuracy (in percentage) of mocap10 classification.

200 factors.

C.3. Detailed Results on mocap10 Classification

We include error bar results for the mocap10 classification task in Table 2.