MOTION-DVAE: UNSUPERVISED LEARNING FOR FAST HUMAN MOTION DENOISING - SUPPLEMENTARY MATERIAL

Anonymous author(s)

1 Motion-DVAE training

1.1 Implementation of the generative model

We recall the generative model of Motion-DVAE:

$$p_\theta(x_{1:T}, z_{1:T}|x_0) = p_\theta(z_1|x_0)p_\theta(x_1|z_1, x_0)\prod_{t=2}^{T} p_\theta(z_t|z_{1:t-1}, x_0)p_\theta(x_t|z_{1:t}, x_0), \tag{1}$$

We implement $p_\theta(x_t|z_{1:t}, x_0) = \mathcal{N}(x_t; \mu_{\theta_x}(z_{1:t}, x_0), Id)$ with:

- $h_0^x = d_{ini}(x_0)$,
- $h_t^x = d_{h^x}(h_{t-1}^x, z_t)$,
- $\mu_{\theta_x}(h_t^x) = d_x(h_t^x)$.

For $p_\theta(z_t|z_{1:t-1}, x_0) = \mathcal{N}(z_t; \mu_{\theta_z}(z_{1:t-1}, x_0), \sigma_{\theta_z}(z_{1:t-1}, x_0))$, we take:

- $h_0^z = d_{ini}(x_0)$
- $h_t^z = d_{h^z}(h_{t-1}^z, z_t)$
- $[\mu_{\theta_z}(h_t^z), \sigma_{\theta_z}(h_t^z)] = d_z(h_t^z)$

d_{h^x} and d_{h^z} are recurrent neural networks and are both implemented with the same LSTM. d_{ini}, d_x, and d_z are MLPs with 2 hidden layers using group normalization with 16 groups and ReLU activations. The latent dimension is 48, while the LSTM hidden state dimension is 1024.

1.2 Implementation of the approximate posterior

The inference model is:

$$q_\phi(z_{1:T}|x_{0:T}) = q_\phi(z_0|x_{0:T})\prod_t q_\phi(z_t|z_{1:t-1}, x_{t:T}, x_0). \tag{2}$$

$q_\phi(z_t|z_{1:t-1}, x_{t:T}, x_0) = \mathcal{N}(z_t; \mu_\phi(z_{1:t-1}, x_{t:T}, x_0), \sigma_\phi(z_{1:t-1}, x_{t:T}, x_0))$ is implemented by:

- $h_t^{glob} = [h_{t-1}^{lat}, h_t^{data}]$,
- $h_0^{lat} = e_{ini}(x_0)$,
- $h_t^{lat} = e_h^{lat}(h_{t-1}^{lat}, z_t)$,
- $h_0^{data} = 0$,
- $h_t^{data} = e_h^{data}(h_{t+1}^{data}, x_t)$,
- $[\mu_\phi(h_t^{glob}), \sigma_\phi(h_t^{glob})] = e_{glob}(h_t^{glob})$.

e_h^{lat} and e_h^{data} are LSTM neural networks, while e_{ini} and e_{glob} are MLPs. Implementations of LSTM and MLP are similar to the generative model.

1.3 Loss function

From Eq. (1) and Eq. (2), following [2], the ELBO of Motion-DVAE is:

$$\mathcal{L}(\theta, \phi; x_{0:T}) = \sum_{t=1}^{T} \mathbb{E}_{q_\phi}[\log p_\theta(x_t|z_{1:t}, x_0)] - \sum_{t=1}^{T} \mathbb{E}_{q_\phi}[D_{KL}(q_\phi(z_t|z_{1:t-1}, x_{t:T}, x_0)||p_\theta(z_t|z_{1:t-1}, x_0))]. \quad (3)$$

We define the loss function of Motion-DVAE:

$$\mathcal{L}^{train}(\theta, \phi; x_{0:T}) = \mathcal{L}_{rec}(\theta, \phi; x_{0:T}) + \mathcal{L}_{KL}(\theta, \phi; x_{0:T}) + \mathcal{L}_{reg}(\theta, \phi; x_{0:T}). \quad (4)$$

The first term can be developed as:

$$\begin{aligned}
\mathcal{L}_{rec}(\theta, \phi; x_{0:T}) &= -\sum_{t=1}^{T} \mathbb{E}_{q_\phi} \left[\log p_\theta(x_t|x_0, z_{1:t})\right] \\
&= -\sum_{t=1}^{T} \mathbb{E}_{q_\phi} \left[\log \mathcal{N}(x_t; \mu_{\theta_x}(z_{1:t}, x_0), Id)\right] \\
&= \sum_{t=1}^{T} \mathbb{E}_{q_\phi} \left[\| \mu_{\theta_x}(z_{1:t}, x_0) - x_t \|_2^2\right].
\end{aligned}$$

This is a reconstruction term between the input and the output of Motion-DVAE.

The second term is a Kullback-Leibler divergence pushing the posterior distribution towards the prior:

$$\mathcal{L}_{KL}(\theta, \phi; x_{0:T}) = \sum_{t=1}^{T} \mathbb{E}_{q_\phi}[D_{KL}(q_\phi(z_t|z_{1:t-1}, x_{t:T}, x_0)||p_\theta(z_t|z_{1:t-1}, x_0))].$$

We also add a regularization term using the SMPL model to enforce the final human mesh to be as close as possible to the original mesh. It consists of a squared reconstruction error on meshes and joints:

$$\mathcal{L}_{reg}(\theta, \phi; x_{0:T}) = \sum_{t=1}^{T} \mathbb{E}_{q_\phi} \left[\| \mathcal{M}_\beta(\mu_{\theta_x}(z_{1:t}, x_0)) - \mathcal{M}_\beta(x_t) \|_2^2\right].$$

1.4 Learning settings

We train Motion-DVAE with sequences of 30 frames from the AMASS [4] dataset, previously downsampled to 30Hz. Then, learning sequences last 1 second. We choose this duration because we argue that even if human motion can last more than 1 second, direct dependencies between poses rarely exceeds 1 second. To ease learning, following [6], we align the first frame of each sequence in the canonical coordinate frame, meaning that translation r_0 and the first two components of root-orient Φ_0 are 0. This enables focusing on learning spatial-temporal dependencies independently from the starting point of the motion, which makes the motion prior more general.

We use batches of 64 sequences and train Motion-DVAE for 200 epochs. Similar to HuMoR [6], we use Adamax [3] with the same settings and learning rate decays. We also use KL-annealing [1] during the first 50 epochs. However, since our model does not use past predictions for current state prediction, we do not need to perform scheduled sampling.

2 Unsupervised learned denoising posterior distribution

Let's recall the joint distribution in the context of motion denoising:

$$p_\theta(y_{0:T}, v_{0:T}, z_{1:T}|\beta, x_0) = p(v_0)p(y_0|x_0, \beta, v_0) \prod_{t=1}^{T} p_\theta(z_t|z_{1:t-1}, x_0)p(v_t)p_\theta(y_t|z_{1:t}, \beta, v_t, x_0). \quad (5)$$

We want to express the posterior distribution $p_\theta(x_0, \beta, z_{1:T}, v_{0:T}|y_{0:T})$:

$$\begin{aligned}
p_\theta(x_0, \beta, z_{1:T}, v_{0:T}|y_{0:T}) &= p(\beta|v_{0:T}, z_{1:T}, x_0, y_{0:T}) p_\theta(v_{0:T}, z_{1:T}, x_0|y_{0:T}) \\
&= p(\beta|v_{0:T}, z_{1:T}, x_0, y_{0:T}) p(v_{0:T}|z_{1:T}, x_0, y_{0:T}) p_\theta(z_{1:T}|x_0, y_{0:T}) p(x_0|y_{0:T}) \\
&= p(\beta|v_{0:T}, z_{1:T}, x_0, y_{0:T}) \prod_{t=1}^{T} \Big[p(v_t|z_{1:t}, y_t, x_0) p_\theta(z_t|z_{1:t-1}, y_{t:T}, x_0) \Big] p(v_0|y_0, x_0) p(x_0|y_{0:T}) \\
&\simeq p(\beta|y_{0:T}) \prod_{t=1}^{T} \Big[p(v_t|z_{1:t}, y_t, x_0) p_\theta(z_t|z_{1:t-1}, y_{t:T}, x_0) \Big] p(v_0|y_0, x_0) p(x_0|y_{0:T}).
\end{aligned}$$

As can be seen in the above calculations, we approximate the posterior of β. We choose to ignore the noise and the latent motion to simplify the model. In practice, β is computed as the average of the observed sequence of body shapes.

As a reminder, we define the following approximate posterior:

$$q(x_0, \beta, z_{1:T}, v_{0:T}|y_{0:T}) = q(\beta|y_{0:T}) \prod_{t=1}^{T} \Big[q_\gamma(v_t|z_{1:t}, y_t, x_0) q_\phi(z_t|z_{1:t-1}, y_{t:T}, x_0) \Big] q_\gamma(v_0|y_0, x_0) q_\omega(x_0|y_{0:T}). \quad (6)$$

3 ELBO derivation and loss functions

For finetuning Motion-DVAE, we aim to minimize the Kullback-Leibler divergence:

$$\min_{\phi,\gamma,\omega} D_{KL}(q_{\phi,\gamma,\omega}(x_0, \beta, z_{1:T}, v_{0:T}|y_{0:T}) || p_\theta(x_0, \beta, z_{1:T}, v_{0:T}|y_{0:T})). \quad (7)$$

We will do it by maximizing the ELBO:

$$\mathcal{L}(\phi, \gamma, \omega) = \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})} \left[\log p_\theta(y_{0:T}, v_{0:T}, z_{1:T}, x_0, \beta) - \log q_{\phi,\gamma,\omega}(x_0, \beta, z_{1:T}, v_{0:T}|y_{0:T}) \right]. \quad (8)$$

3.1 ELBO decomposition

One can notice that the first term of the ELBO involves the joint distribution defined in Eq. (5). Taking the logarithm and expectation of this joint distribution, we obtain:

$$\mathbb{E}_{q_{\phi,\gamma,\omega}} \left[\log p(v_0) + \log p(y_0|x_0, \beta, v_0) \right] + \sum_{t=1}^{T} \mathbb{E}_{q_{\phi,\gamma,\omega}} \left[\log p_\theta(z_t|x_0, z_{1:t-1}) + \log p(v_t) + \log p_\theta(y_t|x_0, z_{1:t}, \beta, v_t) \right].$$

Similarly, the second term involves:

$$q_{\phi,\gamma,\omega}(x_0, \beta, z_{1:T}, v_{0:T}|y_{0:T}) = q(\beta|y_{0:T}) \prod_{t=1}^{T} \Big[q_\gamma(v_t|z_{1:t}, y_t, x_0) q_\phi(z_t|z_{1:t-1}, y_{t:T}, x_0) \Big] q_\gamma(v_0|y_0, x_0) q_\omega(x_0|y_{0:T}),$$

which becomes after taking the logarithm and expectation:

$$\begin{aligned}
&\mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})} \left[\log q(\beta|y_{0:T}) + \log q_\gamma(v_0|y_0, x_0) + \log q_\omega(y_{0:T}) \right] \\
&+ \sum_{t=1}^{T} \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})} \left[\log q_\gamma(v_t|z_{1:t}, y_t, x_0) + \log q_\phi(z_t|z_{1:t-1}, y_{t:T}, x_0) \right].
\end{aligned}$$

The decomposed ELBO is then:

$$\begin{aligned}
&\mathcal{L}(\phi,\gamma,\omega) = \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\left[\log p(y_0|x_0,\beta,v_0) + \sum_{t=1}^{T}\log p_\theta(y_t|z_{1:t},\beta,v_t,x_0)\right] \\
&+ \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\left[-\log q_\omega(x_0|y_{0:T})\right] \\
&+ \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\left[\sum_{t=1}^{T}\log p_\theta(z_t|z_{1:t-1},x_0) - \sum_{t=1}^{T}\log q_\phi(z_t|z_{1:t-1},y_{t:T},x_0)\right] \\
&+ \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\left[\sum_{t=0}^{T}\log p(v_t) - \sum_{t=1}^{T}\log q_\gamma(v_t|z_{1:t},y_t,x_0)\right] \\
&+ \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\left[-\log q(\beta|y_{0:T})\right],
\end{aligned}$$

which can be rewritten as:

$$\begin{aligned}
&\mathcal{L}(\phi,\gamma,\omega) = \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\left[\log p(y_0|x_0,\beta,v_0) + \sum_{t=1}^{T}\log p_\theta(y_t|z_{1:t},\beta,v_t,x_0)\right] \\
&- \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\left[\log q_\omega(x_0|y_{0:T})\right] \\
&- \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,v_{0:T}|z_{1:T},y_{0:T})}\left[D_{KL}(\prod_{t=1}^{T} q_\phi(z_t|z_{1:t-1},y_{t:T},x_0) \parallel \prod_{t=1}^{T} p_\theta(z_t|z_{1:t-1},x_0))\right] \\
&- \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T}|v_{0:T},y_{0:T})}\left[D_{KL}(\prod_{t=0}^{T} q_\gamma(v_t|z_{1:t},y_t,x_0) \parallel \prod_{t=0}^{T} p(v_t))\right] \\
&- \mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\left[\log q(\beta|y_{0:T})\right].
\end{aligned}$$

3.2 ELBO terms calculation

3.2.1 Data commitment term

The first term is a data commitment term. We have:

$$\begin{aligned}
&\mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\left[\log p(y_0|x_0,\beta,v_0) + \sum_{t=1}^{T}\log p_\theta(y_t|x_0,z_{1:t},\beta,v_t)\right] \\
&= \mathbb{E}_{q_{\phi,\gamma,\omega}}\left[\log \mathcal{N}(y_0;\mathcal{M}_\beta(x_0),v_0) + \sum_{t=1}^{T}\log \mathcal{N}(y_t;\mathcal{M}_\beta(\mu_{\theta_x}(x_0,z_{1:t})),v_t))\right] \\
&= \frac{1}{2}\mathbb{E}_{q_{\phi,\gamma,\omega}}\left[\sum_{j,d}\frac{1}{v_{0,j,d}}(y_{0,j,d} - \mathcal{M}_\beta(x_0)_{j,d})^2 + \sum_{t,j,d}\frac{1}{v_{t,j,d}}(y_{t,j,d} - \mathcal{M}_\beta(\mu_{\theta_x}(x_0,z_{1:t}))_{j,d})^2\right]
\end{aligned}$$

As expected, those are Mean Squared Errors weighted by the inverse of the variance.

3.2.2 Initial state predictor term

The second term is linked to the initial state predictor:

$$\mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\left[\log q_\omega(x_0|y_{0:T})\right] = \mathbb{E}_{q_\omega(x_0|y_{0:T})}\left[\log q_\omega(x_0|y_{0:T})\right].$$

This term corresponds to the negative entropy of $q_\omega(x_0|y_{0:T})$. We chose not to use it during learning because it would increase the initial state predicted variance, which is not a desired behavior.

3.2.3 Motion prior term

The third term of the ELBO uses Motion-DVAE to implement a motion prior:

$$
\begin{aligned}
&\mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,v_{0:T}|z_{1:T},y_{0:T})}\left[D_{KL}\left(\prod_{t=1}^{T} q_\phi(z_t|z_{1:t-1},y_{t:T},x_0) \parallel \prod_{t=1}^{T} p_\theta(z_t|z_{1:t-1},x_0)\right)\right] \\
&= \sum_{t=1}^{T} \mathbb{E}_{q_\omega q_\phi}\Big[D_{KL}\big(q_\phi(z_t|z_{1:t-1},y_{t:T},x_0)||p_\theta(z_t|x_0,z_{1:t-1})\big)\Big] \\
&= \sum_{t=1}^{T} \mathbb{E}_{q_\omega q_\phi}\Big[D_{KL}\big(\mathcal{N}(z_t;\mu_\phi(z_{1:t-1},x_{t:T},x_0),\sigma_\phi(z_{1:t-1},x_{t:T},x_0)) \parallel \mathcal{N}(z_t;\mu_{\theta_z}(z_{1:t-1},x_0),\sigma_{\theta_z}(z_{1:t-1},x_0))\big)\Big] \\
&= -\frac{1}{2}\sum_{t=1}^{T} \mathbb{E}_{q_\omega q_\phi}\left[\log\frac{\sigma_{\mu_{\theta_z}}}{\sigma_\phi} - 1 + \frac{\sigma_\phi}{\sigma_{\mu_{\theta_z}}} + \frac{(\mu_{\mu_{\theta_z}}-\mu_\phi)^2}{\sigma_{\mu_{\theta_z}}}\right].
\end{aligned}
$$

3.2.4 Noise prior term

The fourth term of the ELBO is a noise prior term:

$$
\begin{aligned}
&\mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T}|v_{0:T},y_{0:T})}\left[D_{KL}(\prod_{t=0}^{T} q_\gamma(v_t|z_{1:t},y_t,x_0) \parallel \prod_{t=0}^{T} p(v_t)\right] \\
&= \sum_{t=1}^{T} \mathbb{E}_{q_\omega q_\phi}\left[D_{KL}(q_\gamma(v_t|z_{1:t},y_t,x_0) \parallel p(v_t)\right] \\
&= \sum_{t,j,d} \mathbb{E}_{q_\omega q_\phi}\left[D_{KL}\left(\mathcal{IG}\left(v_{t,j,d},\alpha_\gamma(z_{1:t},y_{t,j,d},x_0),\beta_\gamma(z_{1:t},y_{t,j,d},x_0)\right) \parallel \mathcal{IG}\left(v_{t,j,d},\frac{\lambda}{2},\frac{\lambda}{2}\right)\right)\right] \\
&= \sum_{t,j,d} \mathbb{E}_{q_\omega q_\phi}\left[\left(\alpha_\gamma - \frac{\lambda}{2}\right)\psi\left(\alpha_\gamma\right) - \log\Gamma\left(\alpha_\gamma\right) + \log\Gamma\left(\frac{\lambda}{2}\right) + \frac{\lambda}{2}\left(\log\beta_\gamma - \log\frac{\lambda}{2}\right) + \alpha_\gamma\frac{\frac{\lambda}{2}-\beta_\gamma}{\beta_\gamma}\right],
\end{aligned}
$$

where Γ and ψ are the Gamma and Digamma functions.

3.2.5 Body shape term

The last term is about body shape:

$$
\mathbb{E}_{q_{\phi,\gamma,\omega}(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\left[\log q(\beta|y_{0:T})\right] = \mathbb{E}\left[\delta(\beta - \bar{\beta}_{SPIN}(y_{0:T}))\right]
$$

Since that term does not depend on any learned parameter, we ignore it during training.

4 Final predictions from observations

4.1 Initial state

We start by predicting the initial state:

$$
\hat{x}_0^{3d} = \mathbb{E}_{q_\omega(x_0|y_{0:T})}[\mathcal{M}_\beta(x_0)] \simeq \mathcal{M}_\beta(\mu_\omega(y_{0:T})), \tag{9}
$$

where μ_ω is the mean vector of $q_\omega(x_0|y_{0:T})$.

4.2 Motion prediction

Then we need to compute $\hat{x}_{1:T} = \mathbb{E}_{p_\theta(x_{1:T}|y_{0:T})}[x_{1:T}]$ where

$$
\begin{aligned}
p(x_{1:T}|y_{0:T}) &= \int p_\theta(x_{1:T},x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})\; d_{x_0}d_\beta d_{z_{1:t}}d_{v_{0:T}} \\
&= \int p_\theta(x_{1:T}|x_0,\beta,z_{1:T},v_{0:T},y_{0:T})p_\theta(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})\; d_{x_0}d_\beta d_{z_{1:t}}d_{v_{0:T}} \\
&= \mathbb{E}_{p_\theta(x_0,\beta,z_{1:T},v_{0:T}|y_{0:T})}\Big[p_\theta(x_{1:T}|x_0,\beta,z_{1:T},v_{0:T},y_{0:T})\Big].
\end{aligned}
$$

Approximating the posterior $p_\theta(x_0, \beta, z_{1:T}, v_{0:T}|y_{0:T})$ with $q_{\phi,\gamma,\omega}$ we obtain:

$$\hat{x}_{1:T} = \mathbb{E}_{q_{\phi,\gamma,\omega}}\Big[\mathbb{E}_{p_\theta(x_{1:T}|x_0,\beta,z_{1:T},v_{0:T},y_{0:T})}[x_{1:T}]\Big]. \tag{10}$$

We need to calculate $p_\theta(x_{1:T}|x_0, \beta, z_{1:T}, v_{0:T}, y_{0:T})$:

$$\begin{aligned}
\log p_\theta(x_{1:T}|x_0, \beta, z_{1:T}, v_{0:T}, y_{0:T}) &\overset{c}{=} \log p_\theta(x_{1:T}, x_0, \beta, z_{1:T}, v_{0:T}, y_{0:T}) \\
&\overset{c}{=} \sum_t \log p(y_t|x_t, v_t) p_\theta(x_t|z_{1:T}, x_0) \\
&= \sum_t \log \mathcal{N}(y_t; x_t, \mathrm{diag}(v_t)) \mathcal{N}(x_t; \mu_{\theta_x}, \sigma_{\theta_x}),
\end{aligned}$$

where $\overset{c}{=}$ denotes equality up to an additive constant that does not depend on $x_{1:T}$. We can further develop the last line and identify:

$$\log p_\theta(x_{1:T}|x_0, \beta, z_{1:T}, v_{0:T}, y_{0:T}) = \sum_{t,j,d} \log \mathcal{N}\left(x_{t,j,d}; \frac{v_{t,j,d}(\mu_{\theta_x})_{j,d} + (\sigma_{\theta_x})_{j,d} y_{t,j,d}}{v_{t,j,d} + (\sigma_{\theta_x})_{j,d}}, \frac{v_{t,j,d}(\sigma_{\theta_x})_{j,d}}{v_{t,j,d} + (\sigma_{\theta_x})_{j,d}}\right).$$

This finally leads to:

$$\hat{x}^{3d}_{t,j,d} = \mathbb{E}_{q_{\phi,\gamma,\omega}}\left[\frac{v_{t,j,d}\mathcal{M}_\beta(\mu_{\theta_x}(x_0, z_{1:t-1})_{j,d}) + y^{3d}_{t,j,d}}{v_{t,j,d} + 1}\right]. \tag{11}$$

5 Unsupervised denoising learning

5.1 Neural networks implementation

During the unsupervised denoising training, we introduce 2 neural networks, an initial state predictor and a noise predictor. Those 2 models were not necessary for training Motion-DVAE on clean motion capture data since x_0 was known and there was no noise in the observations.

As described in the main paper, the initial state predictor implements $q_\omega(x_0|y_{0:T}) = \mathcal{N}(x_0; \mu_\omega(y_{0:T}), \sigma_\omega(y_{0:T}))$. It is implemented by:

- $h_0 = 0$
- $h_t = r_h(h_{t+1}, y_t)$
- $\hat{x_0} = e_{ini}(h_T)$

r_h is an anticausal LSTM neural network. We want it to take the observations $y_{0:T}$ backward in time because x_0 should depend more on y_0 than on y_T. e_{ini} is an MLP.

For the noise predictor, we implement $q_\gamma(v_t|z_{1:t}, y_t, x_0) = \prod_{j,d} \mathcal{IG}\left(v_{t,j,d}; \alpha_\gamma(z_{1:t}, y_t, x_0), \beta_\gamma(z_{1:t}, y_t, x_0)\right)$ as follows:

- $h^z_0 = d_{ini}(x_0)$
- $h^z_t = d_{h^z}(h^z_{t-1}, z_t)$
- $[\alpha_\gamma, \beta_\gamma] = e_b(h_t, y_t)$

d_{ini} and d_{h^z} are the MLP and the LSTM neural networks previously defined for Motion-DVAE. e_b is an MLP only used by the noise predictor.

5.2 Learning settings

Unsupervised denoising training for regression mode is performed on training data for 500 epochs, with early stopping when the validation loss does not improve for 10 epochs. Note that for noisy AMASS [4] data, the training converges in about 50 epochs only, probably due to a large amount of training data. In optimization mode, we fix the number of iterations (200 iterations on i3DB [5], and 50 for AMASS).

During unsupervised denoising learning, the decoder and prior networks are fixed, preserving the motion prior. We optimize the weights of the encoder, initial state predictor, and noise predictor networks. As for training the original Motion-DVAE, we use KL-annealing [1].