# Group Meeting - March 12, 2021

Paper review & Research progress

Truong Son Hy [*]

[*]Department of Computer Science
The University of Chicago

Ryerson Physical Lab

## Content

- Literature review:
  - **Energy-Based Processes for Exchangeable Data**, `https://arxiv.org/abs/2003.07521`
- Research update:
  - Conditional contrastive generation for 3D structure: MD17, ShapeNet.

## Paper 1

**Energy-Based Processes for Exchangeable Data**
Mengjiao Yang, Bo Dai, Hanjun Dai, Dale Schuurmans
`https://arxiv.org/abs/2003.07521`

## Introduction (1)

### Two key considerations

Modeling a distribution over a space of instances, where each instance is an **unordered set** of elements involves:

1. the elements within a single instance are **exchangeable** (i.e. the elements are order invariant).
2. the cardinalities of the instances (sets) **vary** (i.e. instances don't need to have the same cardinality).

## Introduction (2)

### Autoregressive approach

Use RNNs to model distributions over instances $\boldsymbol{x} = \{x_1, .., x_n\}$ without assuming fixed cardinality in an **autoregressive** manner:

$$p(\boldsymbol{x}) = \prod_{i=1}^{n} p(x_i | x_{1:i-1})$$

for a permutation of its elements.

### Exchangeability

RNNs has been empirically successful, but does not respect **exchangeability**.

## Introduction (3)

### De Finetti's theorem

To explicitly ensure exchangeability, we exploits the **De Finetti's theorem**, which assures us that for any distribution over **exchangeable** elements $\boldsymbol{x} = \{x_1, .., x_n\}$ the instance probability can be decomposed as:

$$p(\boldsymbol{x}) = \int \prod_{i=1}^{n} p(x_i|\theta)p(\theta)d\theta$$

for some latent variable $\theta$.

### Proposal

**Energy-Based Processes** $\mathcal{EBP}s$:

- Combines energy-based models (EBMs) and stochastic processes.
- Obtains exchangeability and varying-cardinality.
- **Neural Collapsed Inference** (NCI) to train $\mathcal{EBP}s$.

## Energy-based models

### EBM

An EBM over $\Omega \subset \mathbb{R}^d$ with fixed dimension $d$ is defined as:

$$p_f(x) = \exp(f(x) - \log Z(f)) = \frac{1}{Z(f)} \exp(f(x))$$

for $x \in \Omega$, where $f(x) : \Omega \to \mathbb{R}$ is the energy function, and:

$$Z(f) = \int_\Omega \exp(f(x))dx$$

is the partition function. Let: $\mathcal{F} = \{f(\cdot) : Z(f) < \infty\}$.

### Adversarial dynamics embedding (ADE)

$$\max_f \min_{q(x,v)\in\mathcal{P}} \hat{\mathbb{E}}[f(x)] - H(q(x,v)) - \mathbb{E}_{q(x,v)}\left[f(x) - \frac{\lambda}{2} v^T v\right]$$

## Stochastic Processes (1)

### Stochastic Processes

Consider a stochastic process given by a collection of random variables $\{X_t; t \in \mathcal{T}\}$ indexed by $t$, where the marginal distribution for any finite set of indices $\{t_1, .., t_n\} \in \mathcal{T}$ (without order) is specified:

$$p(x_{t_1:t_n}) = p(x_{t_1}, .., x_{t_n} | \{t_i\}_{i=1}^{n})$$

## Stochastic Processes (2)

**Kolmogorov extension theorem**

Sufficient conditions for designing a valid stochastic processes:

- **Exchangeability:** The marginal distribution for any finite set of random variables is **invariant to permutation** order.

$$p(x_{t_1}, .., x_{t_n} | \{t_i\}_{i=1}^{n}) = p(\pi(x_{t_{1:t_n}}) | \pi(\{t_i\}_{i=1}^{n}))$$

  where $p(\pi(x_{t_{1:t_n}})) = p(x_{\pi(t_1)}, .., x_{\pi(t_n)})$.

- **Consistency:** The partial marginal distribution, obtained by marginalizing additional variables in the finite sequence, is the same as the one obtained from the original infinite sequence. For $n \geq m \geq 1$:

$$p(x_{t_{1:t_m}} | \{t_i\}_{i=1}^{m}) = \int p(x_{t_{1:t_n}} | \{t_i\}_{i=1}^{n}) dx_{t_{m+1:t_n}}$$

# Stochastic Processes (3)

## Representation of Stochastic Processes

For any stochastic process $(x_{t_1}, x_{t_2}, ..) \sim \mathcal{SP}$ that can be constructed via Kolmogorov extension theorem, the process can be equivalently represented by a latent variable model:

$$\theta \sim p(\theta), \qquad x_{t_i} \sim p(x|\theta, t_i), \forall i \in \{1, .., n\} \forall n$$

where $\theta$ can be finite or infinite dimensional.

## $\mathcal{EBP}$ Construction (1)

Let the energy function $f$ parameterized by learnable $w$:

$$p_w(x|\theta, t) = \frac{\exp(f_w(x, t; \theta))}{Z(f_w, t; \theta)}$$

where $Z(f_w, t; \theta) = \int \exp(f_w(x, t; \theta))dx$.

### $\mathcal{EBP}$ on arbitrary finite marginals

$$p_w(x_{t_1:t_n}|\{t_i\}_{i=1}^n) = \int \frac{\exp\left(\sum_{i=1}^n f_w(x_{t_i}, t_i; \theta)\right)}{Z(f_w, t; \theta)} p(\theta)d\theta$$

## $\mathcal{EBP}$ Construction (2)

### Gaussian Processes

$$\theta \sim \mathcal{N}(0, I_d)$$

$$f_w(x, t; \theta) = -\frac{1}{2\sigma^2}||x - \theta^T \phi(t)||^2$$

where $w = \{\sigma, \phi(\cdot)\}$, with $\phi(\cdot)$ denotes the feature mapping. Let $k(t, t') = \phi(t)^T \phi(t')$, the marginalized distribution:

$$p(x_{t_1:t_n} | \{t_i\}_{i=1}^n) = \mathcal{N}(0, K(t_{1:n}) + \sigma^2 I_n)$$

where $K(t_{1:n}) = [k(t_i, t_j)]_{i,j}^n$.

$$X_t \sim \mathcal{GP}(0, K(t_{1:n}) + \sigma^2 I_n)$$

# $\mathcal{EBP}$ Construction (3)

## Neural Processes

Neural processes ($\mathcal{NP}$s) are explicitly defined by a latent variable model:

$$p(x_{t_1:t_n} | \{t_i\}_{i=1}^n) = \int \prod_{i=1}^n \mathcal{N}(x | h_w(t_i;\theta)) p(\theta) d\theta$$

where $h_w(\cdot;\theta)$ is a neural network.

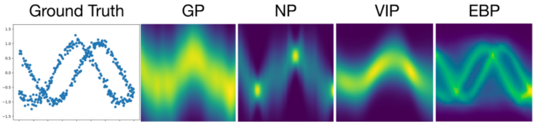


Figure 1: The ground truth data and learned energy functions of $\mathcal{GP}$, $\mathcal{NP}$, $\mathcal{VIP}$, and $\mathcal{EBP}$ (from left to right). $\mathcal{EBP}$ successfully captures multi-modality of the toy data as GP and NP exhibiting only a single mode; see Section 5 for details.

## Unconditional $\mathcal{EBP}$s Extension (1)

When the indices $\{t_i\}_{i=1}^n$ are not observed:

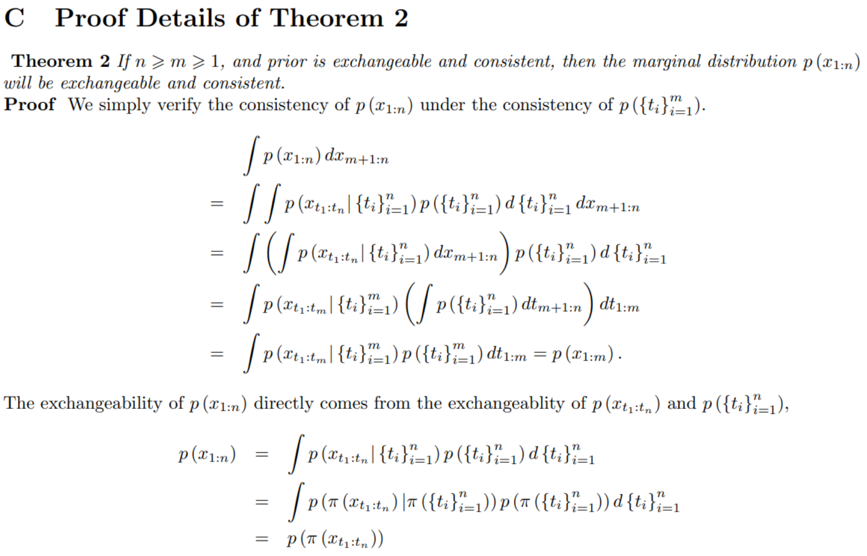
$$p_w(x_{1:n}) = \int p_w(x_{t_1:t_n} | \{t_i\}_{i=1}^n) p(\{t_i\}_{i=1}^n) dt_{1:n}$$

$$= \int p_w(x_{t_1:t_n} | \{t_i\}_{i=1}^n, \theta) p(\theta) p(\{t_i\}_{i=1}^n) d\theta dt_{1:n}$$

**Theorem 2**

If $n \geq m \geq 1$, and the prior is exchangeable and consistent, then the marginal distribution $p(x_{1:n})$ will be exchangeable and consistent.

## Unconditional $\mathcal{EBP}$s Extension (2)



### C Proof Details of Theorem 2

**Theorem 2** *If $n \geqslant m \geqslant 1$, and prior is exchangeable and consistent, then the marginal distribution $p(x_{1:n})$ will be exchangeable and consistent.*

**Proof** We simply verify the consistency of $p(x_{1:n})$ under the consistency of $p(\{t_i\}_{i=1}^m)$.

$$\begin{aligned} & \int p(x_{1:n})\, dx_{m+1:n} \\ = & \int\int p(x_{t_1:t_n} | \{t_i\}_{i=1}^n)\, p(\{t_i\}_{i=1}^n)\, d\{t_i\}_{i=1}^n\, dx_{m+1:n} \\ = & \int \left( \int p(x_{t_1:t_n} | \{t_i\}_{i=1}^n)\, dx_{m+1:n} \right) p(\{t_i\}_{i=1}^n)\, d\{t_i\}_{i=1}^n \\ = & \int p(x_{t_1:t_m} | \{t_i\}_{i=1}^m) \left( \int p(\{t_i\}_{i=1}^n)\, dt_{m+1:n} \right) dt_{1:m} \\ = & \int p(x_{t_1:t_m} | \{t_i\}_{i=1}^m)\, p(\{t_i\}_{i=1}^m)\, dt_{1:m} = p(x_{1:m}). \end{aligned}$$

The exchangeability of $p(x_{1:n})$ directly comes from the exchangeablity of $p(x_{t_1:t_n})$ and $p(\{t_i\}_{i=1}^n)$,

$$\begin{aligned} p(x_{1:n}) &= \int p(x_{t_1:t_n} | \{t_i\}_{i=1}^n)\, p(\{t_i\}_{i=1}^n)\, d\{t_i\}_{i=1}^n \\ &= \int p(\pi(x_{t_1:t_n}) | \pi(\{t_i\}_{i=1}^n))\, p(\pi(\{t_i\}_{i=1}^n))\, d\{t_i\}_{i=1}^n \\ &= p(\pi(x_{t_1:t_n})) \end{aligned}$$

# Neural Collapsed Reparameterization (1)

Given samples $\mathcal{D} = \{x_{1:n}^i\}_{i=1}^N$:

$$\max_w \hat{\mathbb{E}}_{\mathcal{D}}[\log p_w(x_{1:n})]$$

Integrations that are not tractable:

1. The partition function

$$Z(f_w, t, \theta) = \int \exp(f_w(x, t; \theta) dx$$

   where $f_w(x, t; \theta)$ is a parameterized neural net.

2. The integration over $\theta$ for:

$$p_w(x_{t_1:t_n} | \{t_i\}_{i=1}^n) = \int \frac{\exp\left(\sum_{i=1}^n f_w(x_{t_i}, t_i; \theta)\right)}{Z(f_w, t; \theta)} p(\theta) d\theta$$

3. Integration over $\{t_i\}_{i=1}^n$ for:

$$p_w(x_{1:n}) = \int p_w(x_{t_1:t_n} | \{t_i\}_{i=1}^n) p(\{t_i\}_{i=1}^n) dt_{1:n}$$

## Neural Collapsed Reparameterization (2)

**Algorithm 1** Neural Collapsed Inference

```
1: Initialize $W_1$ randomly, set length of steps $T$.
2: for iteration $k = 1, \dots, K$ do
3:     Sample mini-batch $\left\{x^j_{1:n_j}\right\}_{j=1}^{b}$ from dataset $\mathcal{D}$.
4:     Sample $\theta^j \sim q_\alpha(\theta|x_{1:n})$, $\forall j = 1, \dots, b$.
5:     Sample $\tilde{x}^j_{1:n}, \tilde{v}^j \sim q_\beta(x_{1:n}, v|\theta)$, $\forall j = 1, \dots, b$.
6:     $\{\beta_{k+1}\} = \beta_k - \gamma_k \hat{\nabla}_\beta L(\alpha_k, \beta_k, w'_k)$
7:     $\{\alpha, w'\}_{k+1} = \{\alpha, w'\}_k + \gamma_k \hat{\nabla}_{\{\alpha, w'\}} L(\alpha_k, \beta_k; w'_k)$.
8: end for
```

## Experiments (1)

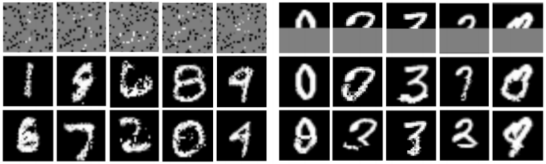


Figure 2: Image completion on MNIST. The first row shows the unobserved pixels in gray and observed pixels in black and white. The second and third rows are two different generated samples given the observed pixels from the first row. Generations are based on randomly selected pixels or the top half of an image.

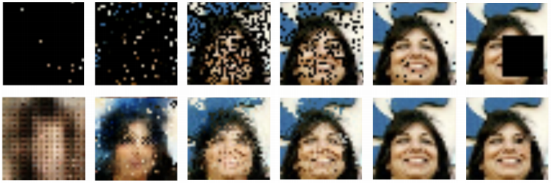


Figure 3: Image completion on CelebA. The first row shows the unobserved pixels in black with an increasing number of observed pixels from left to right (column 1-5). The second row shows the completed image given the observed pixels from the first row.

## Experiments (2)

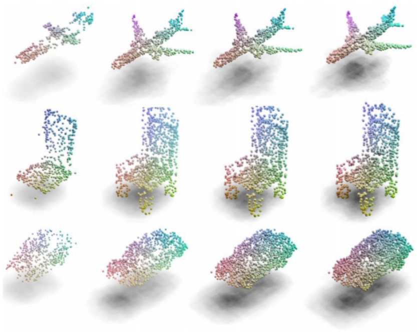


Figure 4: Example point clouds of airplane, chair, and car generated from the learned model.

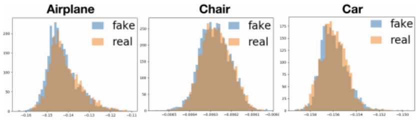


Figure 5: Energy distributions of the generated samples (fake) and training data (real). x-axis is the energy value and y axis is the count of examples. The energy distributions of the generated and real point clouds show significant overlap.

## Experiments (3)

Table 1: Generation results. ↑: the higher the better. ↓: the lower the better. The best scores are highlighted in bold. JSD is scaled by $10^2$, MMD-CD by $10^3$, and MMD-EMD by $10^2$. Each number for l-GAN is from the model trained using either CD or EMD loss, whichever one is better.

| | | JSD (↓) | MMD (↓) | | COV (%, ↑) | |
|---|---|---|---|---|---|---|
| Category | Model | | CD | EMD | CD | EMD |
| Airplane | l-GAN | **3.61** | 0.239 | 3.29 | 47.90 | 50.62 |
| | PC-GAN | 4.63 | 0.287 | 3.57 | 36.46 | 40.94 |
| | PointFlow | 4.92 | **0.217** | 3.24 | 46.91 | 48.40 |
| | EBP (ours) | 3.92 | 0.240 | **3.22** | **49.38** | **51.60** |
| Chair | l-GAN | 2.27 | 2.46 | **7.85** | 41.39 | 41.69 |
| | PC-GAN | 3.90 | 2.75 | 8.20 | 36.50 | 38.98 |
| | PointFlow | 1.74 | **2.42** | 7.87 | 46.83 | 46.98 |
| | EBP (ours) | **1.53** | 2.59 | 7.92 | **47.73** | **49.84** |
| Car | l-GAN | 2.21 | 1.48 | 5.43 | 39.20 | 39.77 |
| | PC-GAN | 5.85 | 1.12 | 5.83 | 23.56 | 30.29 |
| | PointFlow | 0.87 | **0.91** | **5.22** | 44.03 | 46.59 |
| | EBP (ours) | **0.78** | 0.95 | 5.24 | **51.99** | **51.70** |

## Experiments (4)

Table 2: Classification accuracy on ModelNet40. Models are pre-trained on ShapeNet before extracting features on ModelNet40. Linear SVMs are then trained using the learned representations.

| Model | Accuracy |
|---|---|
| VConv-DAE (Sharma et al., 2016) | 75.5 |
| 3D-GAN (Wu et al., 2016) | 83.3 |
| l-GAN (EMD) (Achlioptas et al., 2017) | 84.0 |
| l-GAN (CD) (Achlioptas et al., 2017) | 84.5 |
| PointGrow (Sun et al., 2018) | 85.7 |
| MRTNet-VAE (Gadelha et al., 2018) | 86.4 |
| PointFlow (Yang et al., 2019) | 86.8 |
| PC-GAN (Li et al., 2018) | 87.8 |
| FoldingNet (Yang et al., 2018) | **88.4** |
| EBP (ours) | 88.3 |

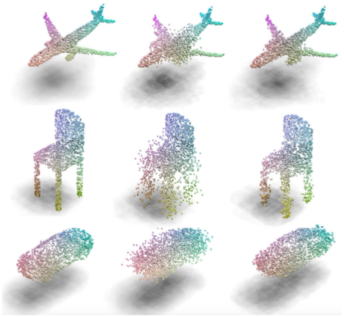


Figure 6: Examples of point cloud denoising using MCMC sampling. From left to right: original, perturbed, and denoised point clouds.