ICML
International Conference
On Machine Learning
ICML 2024

Latent Optimal Paths by Gumbel Propagation for Variational Bayesian Dynamic Programming

Xinlei Niu[1], Christian Walder[2], Jing Zhang[1], Charles Patrick Martin[1]

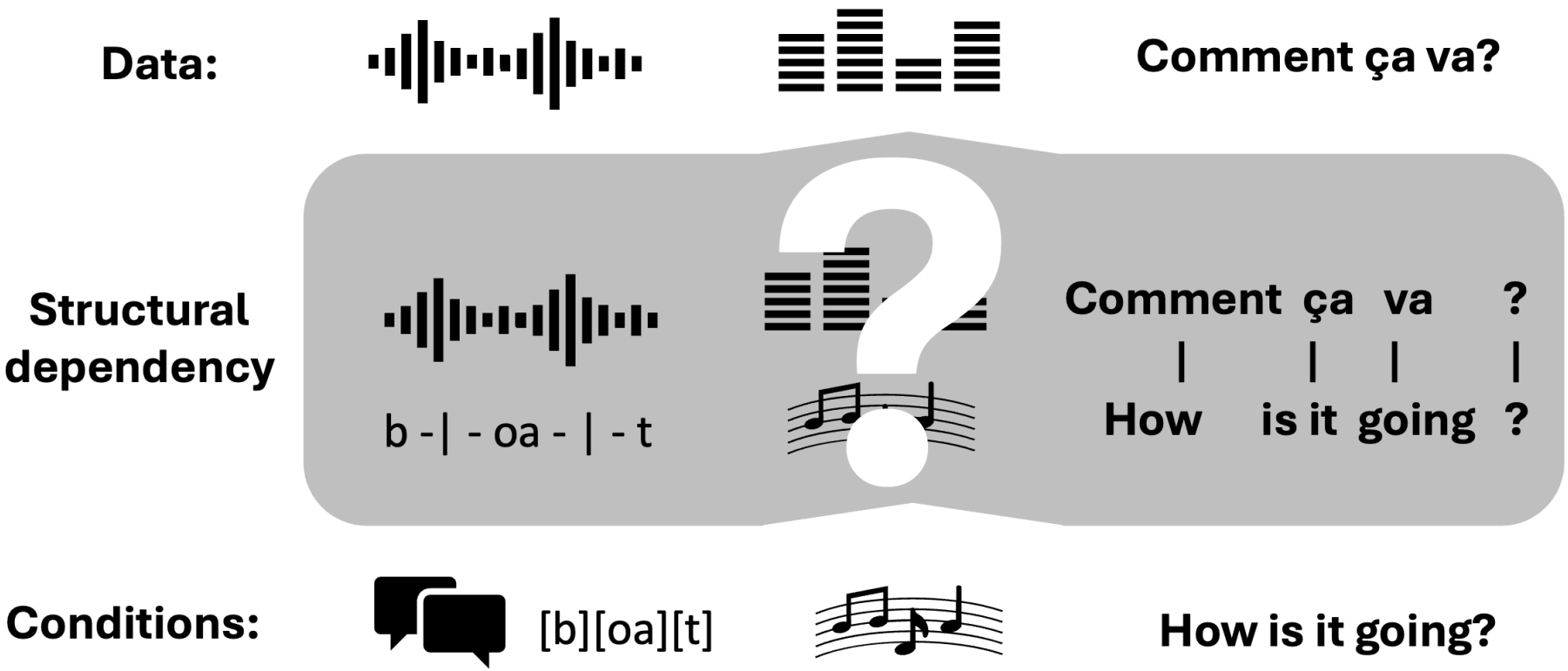
[1]Australian National University

[2]Google DeepMind

Australian
National
University

Google DeepMind

Optimal Paths in Generative Tasks



Related Work

- Traditional optimal path problem
  1. Loose the sparsity of optimal paths with differentiable dynamic programming (DP) (Verdu & Poor, 1987; Kim et al., 2017; Cuturi & Blondel, 2017).
  2. Involve a convex optimization problem (Amos & Kolter, 2017; Djolonga & Krause, 2017).
  3. Maintain the sparsity but smoothed with a convex regularizer (Mensch & Blondel, 2018)
- Downstream generative tasks with unobserved structural dependency
  1. Multiple training strategies, strongly relies on external components (Ren et al.,2020; Liu et al., 2022).
  2. Have a strong assumption about the model structure (Rabiner, 1989; Petrov & Klein, 2007).
  3. Cannot ensure train and test consistency (Kim et al., 2020) or soft approximation (Shen et al.,2018; Lu et al., 2021).

Our Motivation: Stochastic Optimal Paths

Definition

For every possible path $\mathbf{y} \in \mathcal{Y}$, denote by

$$\mathcal{D}(\mathcal{R}, \mathbf{W}, \alpha) \tag{1}$$

Gibbs distribution over a path $\mathbf{y} \in \mathcal{Y}(1, N)$ with probability mass function

$$\mathcal{D}(\mathbf{y}|\mathcal{R}, \mathbf{W}, \alpha) = \frac{\exp(\alpha \|\mathbf{y}\|_{\mathbf{W}})}{\sum_{\widehat{\mathbf{y}} \in \mathcal{Y}(1,N)} \exp(\alpha \|\widehat{\mathbf{y}}\|_{\mathbf{W}})}. \tag{2}$$

1. A **unified** method to obtain structured **sparse** optimal paths.
2. Provide **tractable** closing forms for all ingredients of variational Bayesian inference to capture stochastic optimal paths.
3. Allow **gradient optimization** for probabilistic generative models.

Definition of a Directed Acyclic Graph

- Denote $\mathcal{R} = (\mathcal{V}, \mathcal{E})$ be a directed acyclic graph (DAG) with nodes $\mathcal{V}$ and edges $\mathcal{E}$.
- The nodes are numbered in topological order, s.t., $\mathcal{V} = (1, 2, ..., N)$ and $u < v$ for all $(u, v) \in \mathcal{E}$.
- Assume 1 is the only node without parents and N is the only node without children.
- Denote the edge weights matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ with $w_{i,j} = -\infty$ for all $(u, v) \notin \mathcal{E}$.
- Let $\mathcal{Y}(1, v)$ be the set of all paths from nodes $(1, .., v)$.
- Each path $\mathbf{y} = (y_1, ..., y_{|\mathbf{y}|})$ has a score by summing edge weights along the path, defined as $||\mathbf{y}||_{\mathbf{W}} = \sum_{(u,v) \in \mathbf{y}} w_{u,v}$.

Gumbel Random Variable

Let $\mathcal{G}(\mu)$ denote the unit scale Gumbel random variable with location parameter μ and probability density function.

$$\mathcal{G}(x|\mu) = \exp(-(x-\mu) - \exp(x-\mu)). \tag{3}$$

- Shifting: Let $X \sim \mathcal{G}(\mu)$.

$$X + \text{const.} \sim \mathcal{G}(\mu + \text{const.}). \tag{4}$$

- Max: Let $X_i \sim \mathcal{G}(\mu_i)$ for all $i \in \{1, 2, \ldots, m\}$.

$$\max(\{X_1, X_2, \ldots, X_m\}) \sim \mathcal{G}(\log \sum_{i=1}^{m} \exp(\mu_i)). \tag{5}$$

- Argmax:

$$p(k = \underset{i \in \{1,2,\ldots,m\}}{\text{argmax}} \; X_i) = \frac{\exp(\mu_k)}{\sum_{i=1}^{m} \exp(\mu_i)}. \tag{6}$$

Gumbel Propagation

Gumbel propagation offers an equivalent formulation of the definition that lends itself to dynamic programming by Gumbel max and shift properties.

Lemma

Let

$$Y = \underset{\mathbf{y} \in \mathcal{Y}(1,N)}{\operatorname{argmax}} \{\Omega_{\mathbf{y}}\}, \tag{7}$$

where for all $\mathbf{y} \in \mathcal{Y}(1, N)$,

$$\Omega_{\mathbf{y}} = \alpha \|\mathbf{y}\|_{\mathbf{W}} + G_{\mathbf{y}} \tag{8}$$

$$G_{\mathbf{y}} \sim \mathcal{G}(0). \tag{9}$$

Then the probability of $Y = \mathbf{y}$ *is given by* (2).

Gumbel Propagation (Cont.)

Let the definitions of $\Omega_{\mathbf{y}}$ and $G_{\mathbf{y}}$ extend to all $\mathbf{y} \in \bigcup_{u=1}^{N} \mathcal{Y}(1, u)$, which is the set of all partial paths. We define for each node $v \in \mathcal{V}$ the real-valued random variable

$$Q_v = \max_{\mathbf{y} \in \mathcal{Y}(1,v)} \{\Omega_{\mathbf{y}}\}. \tag{10}$$

Lemma

The Q_v are Gumbel distributed with

$$Q_v \sim \mathcal{G}(\mu_v), \tag{11}$$

where

$$\mu_v = \log \sum_{\mathbf{y} \in \mathcal{Y}(1,v)} \exp(\alpha \, \|\mathbf{y}\|_{\mathbf{W}}). \tag{12}$$

Gumbel Propagation (Cont.)

We now state our main result:

Lemma

The location parameters μ_v satisfy the recursion

$$\mu_1 = 0 \tag{13}$$

$$\mu_v = \log \sum_{u \in \mathcal{P}(v)} \exp(\mu_u + \alpha\, w_{u,v}). \tag{14}$$

for all $v \in \{2, 3, \ldots, N\}$.

Gumbel Propagation (Cont.)

Lemma

Let paths $\mathbf{y} = (y_1, y_2, \ldots, y_{|\mathbf{y}|})$ *denote the component of the random variable* Y *defined in* (7)*, given that* $Y = (y_1, y_2, \ldots, y_{|\mathbf{y}|})$*. The probability of the transition* $v \to u$ *is*

$$\pi_{u,v} \equiv p(y_{i-1} = u | y_i = v, u \in \mathcal{P}(v)) \tag{15}$$

$$= \frac{\exp(\mu_u + \alpha\, w_{u,v})}{\exp(\mu_v)}, \tag{16}$$

for all $i \in \{2, 3, \ldots, N\}$.

Bayesian Dynamic Programming

- Sampling

Corollary

Paths $\mathbf{y} \sim \mathcal{D}(\mathcal{R}, \mathbf{W}, \alpha)$ *may be sampled (in reverse) by*

1. *Initializing* $v = N$,
2. *sampling* $u \in \mathcal{P}(v)$ *with probability* $\pi_{u,v}$,
3. *setting* $v \leftarrow u$,
4. *if* $v = 1$ *then stop, otherwise return to step 2.*

- Likelihood
- KL Divergence within the distribution family of $\mathcal{D}(\mathcal{R}, \cdot, \alpha)$.

Bayesian Dynamic Programming (Cont.)

- Sampling
- Likelihood

Corollary

The path probability may be written

$$\mathcal{D}(\mathbf{y}|\mathcal{R}, \mathbf{W}, \alpha) = \prod_{(u,v)\in\mathbf{y}} \pi_{u,v}. \tag{17}$$

- KL Divergence within the distribution family of $\mathcal{D}(\mathcal{R}, \cdot, \alpha)$.

Bayesian Dynamic Programming (Cont.)

- Sampling
- Likelihood
- KL Divergence within the distribution family of $\mathcal{D}(\mathcal{R}, \cdot, \alpha)$.

Lemma

The KL divergence within the family $\mathcal{D}(\mathcal{R}, \cdot, \alpha)$ is

$$\begin{aligned} &\mathcal{D}_{\mathrm{KL}}\left[\mathcal{D}(\mathcal{R}, \mathbf{W}, \alpha) \middle\| \mathcal{D}(\mathcal{R}, \mathbf{W}^{(r)}, \alpha)\right] \\ &= \mu_N^{(r)} - \mu_N + \alpha \sum_{(u,v)\in\mathcal{E}} \omega_{u,v}\left(w_{u,v} - w_{u,v}^{(r)}\right), \end{aligned} \tag{18}$$

where $\omega_{u,v}$ is the marginal probability of edge (u, v) on $\mathcal{D}(\mathcal{R}, \mathbf{W}, \alpha)$, $\mu_N^{(r)}$ is similar to μ_N but defined in terms of $\mathbf{W}^{(r)}$ rather than $\mathbf{W}$.

BDP-VAE

Find unobserved dependencies between condition and data in a latent space, thereby, helps a better reconstruction [1].

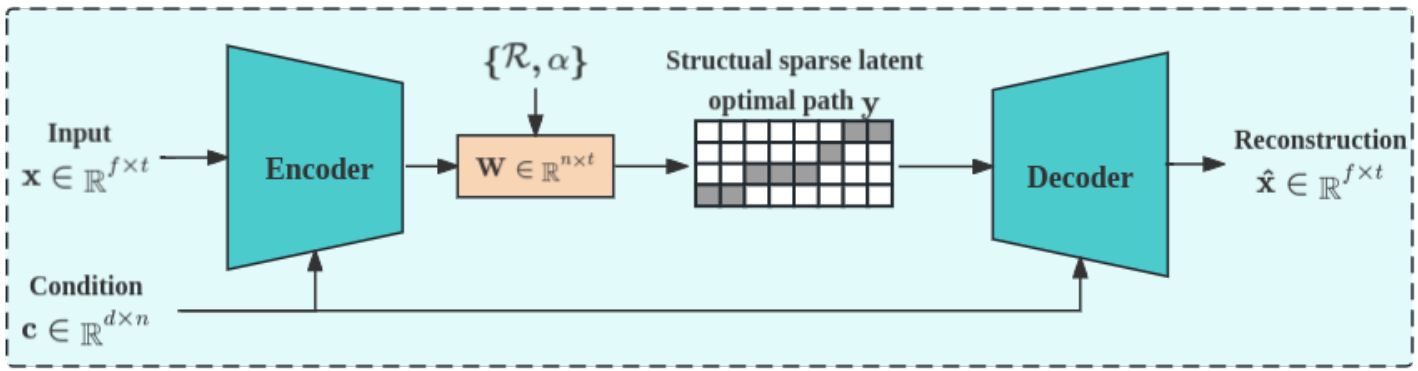


Figure: A pipeline of BDP-VAE, where we wish to find an unobserved hard structural relationship between **x** and **c** in the latent space of VAEs denoted as **y**.

[1] For soft optimal paths, please refer to our discussion in Appendix.D

Application on Downstream Tasks

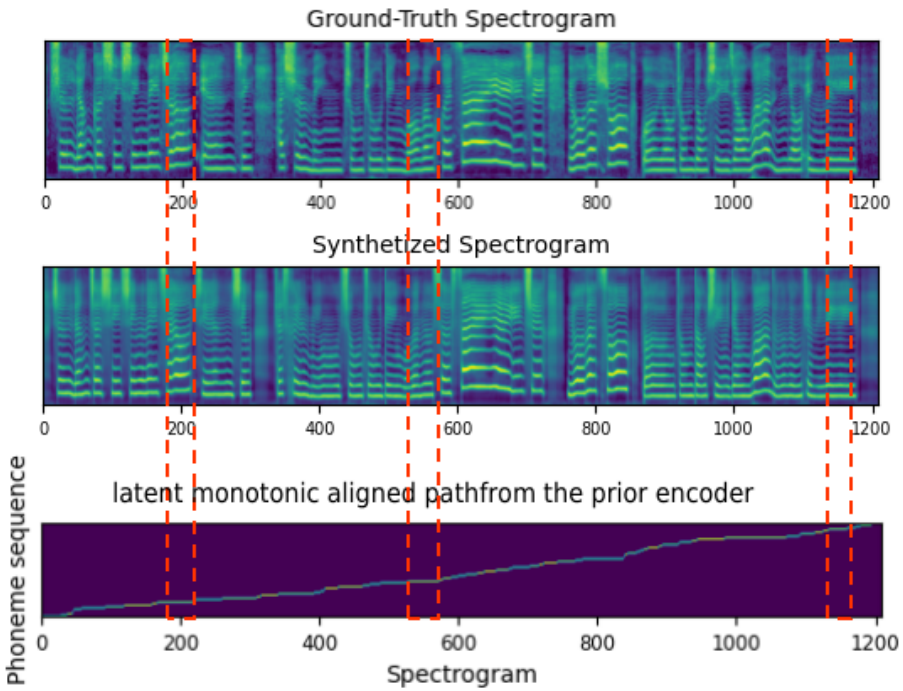
1. **End-to-end Text-to-Speech (TTS)**
   - We adapt a non end-to-end TTS model (FastSpeech2) to BDP-VAE framework, thereby performs an end-to-end TTS model (i.e., BDPVAE-TTS).
   - BDPVAE-TTS outperforms than the baseline method with an end-to-end pipeline.
   - BDPVAE-TTS ensures train and test consistency on finding the monotonic alignment path.

Table: Mel Cepstral Distortion (MCD) and Real-Time Factor (RTF) compared with other TTS models.

Model	**Training**	**Align.(Train)**	**Align. (Infer)**	**MCD**	**RTF**
FastSpeech2 (Baseline)	Non end-to-end	Discrete	Continuous	9.96 ± 1.01	3.87×10^{-4}
Tacotron2	End-to-end	Continuous	Continuous	11.39 ± 1.95	6.07×10^{-4}
VAENAR-TTS	End-to-end	Continuous	Continuous	8.18 ± 0.87	1.10×10^{-4}
Glow-TTS	End-to-end	Discrete	Continuous	8.58 ± 0.89	2.87×10^{-4}
BDPVAE-TTS (ours)	End-to-end	Discrete	Discrete	8.49 ± 0.96	3.00×10^{-4}

Application on Downstream Tasks

1. End-to-end Text-to-Speech (TTS)
2. **End-to-end Singing Voice Synthesis**



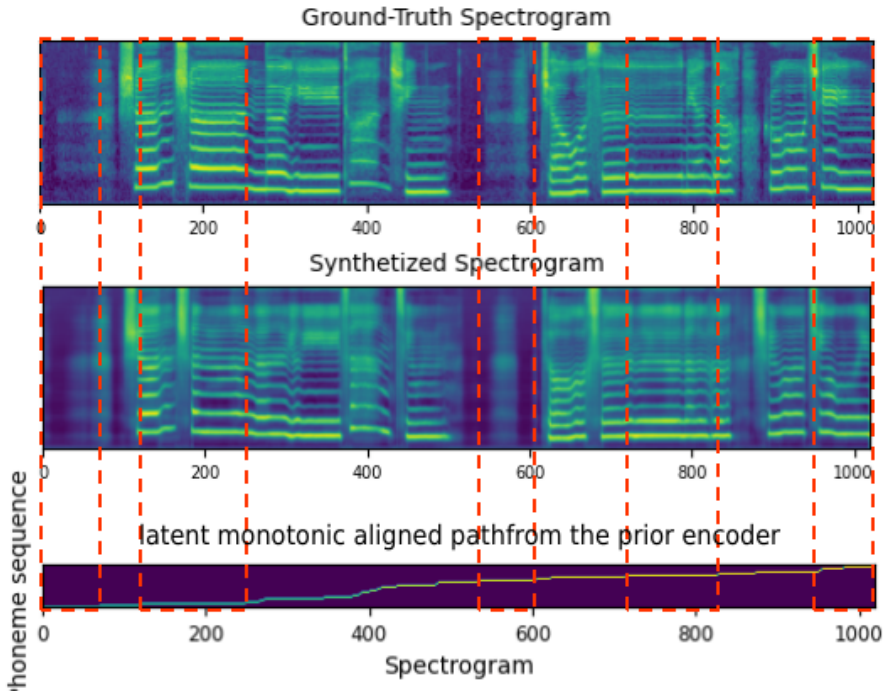


Figure: Visualization of GT, synthesized singing voice spectrogram, and latent optimal path from the prior encoder. *The GT and generated spectrogram are almost identical, and the generated spectrogram has a similar temporal structure to the inferred latent optimal path.*

Verify Behaviour of Latent Optimal Path in BDP-VAE

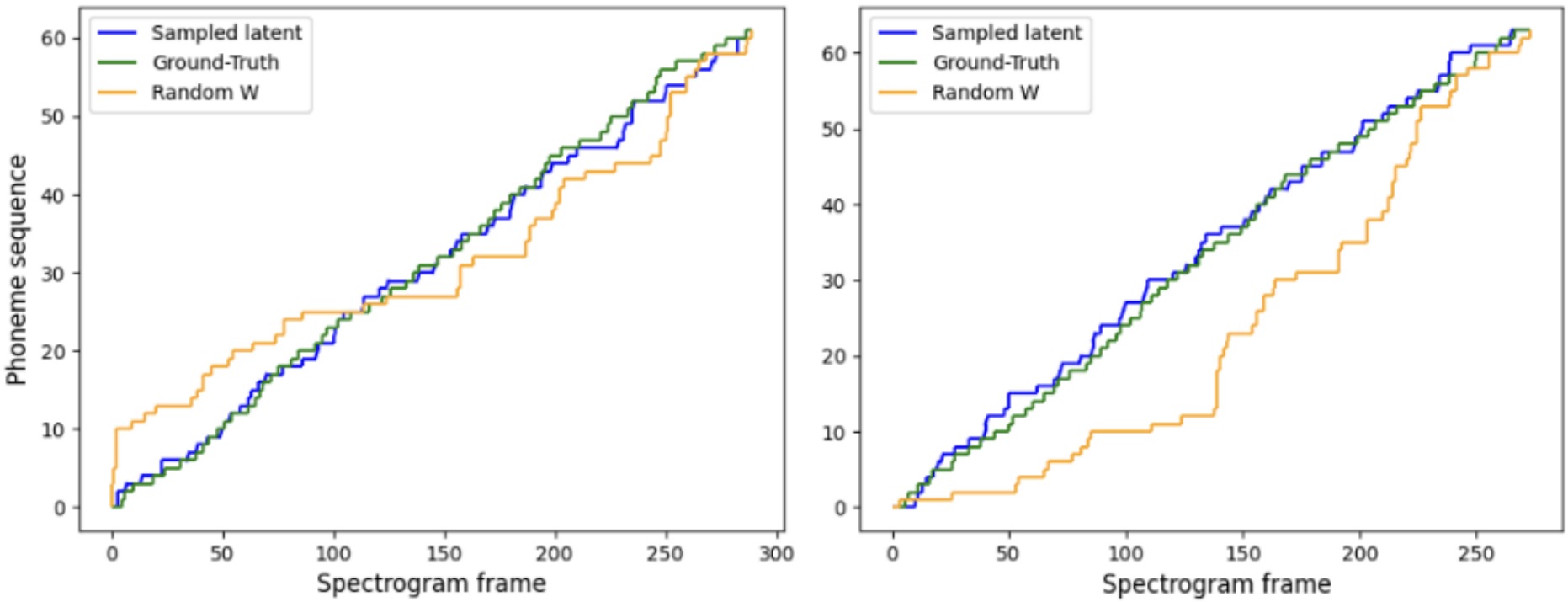


Figure: Visualization of GT alignment between phoneme tokens and spectrogram frames, latent optimal paths from the encoder, optimal paths from random latent space for two audio clips. *BDP-VAE achieves closer alignments with GT, indicating its effectiveness in finding latent optimal paths.*

Latent Optimal Paths by Gumbel Propagation for Variational Bayesian Dynamic Programming

Xinlei Niu[1], Christian Walder[2], Jing Zhang[1], Charles Patrick Martin[1]

[1]Australian National University

[2]Google DeepMind

Our Code

Our Paper