Variational Autoencoder

References:

- 变分推断之傻瓜式推导ELBO
- Auto-Encoding Variational Bayes
- 从极大似然估计到变分自编码器 - VAE 公式推导

Evidence Lower Bound

Suppose we have observation X and latent variable Z. We are interested in the posterior distribution $p(\mathbf{z}|\mathbf{x})$. It is often intractable to compute $p(\mathbf{z}|\mathbf{x})$ due to the dimensionality. Hence, we introduce another distribution $q(\mathbf{z})$ to approximate the true posterior. To minimize the difference between $p(\mathbf{z}|\mathbf{x})$ and $q(\mathbf{z})$, we use the KL divergence:

$$
\begin{aligned}
D_{\mathrm{KL}}\Big(q(\mathbf{z})\|p(\mathbf{z}|\mathbf{x})\Big) &= \int q(\mathbf{z}) \log \frac{q(\mathbf{z})}{p(\mathbf{z}|\mathbf{x})} d\mathbf{z} \\
&= \int q(\mathbf{z}) \log q(\mathbf{z}) d\mathbf{z} - \int q(\mathbf{z}) \log p(\mathbf{z}|\mathbf{x}) d\mathbf{z} \\
&= \int q(\mathbf{z}) \log q(\mathbf{z}) d\mathbf{z} - \int q(\mathbf{z}) \log \frac{p(\mathbf{x},\mathbf{z})}{p(\mathbf{x})} d\mathbf{z} \\
&= \int q(\mathbf{z}) \log q(\mathbf{z}) d\mathbf{z} - \int q(\mathbf{z}) \log p(\mathbf{x},\mathbf{z}) d\mathbf{z} + \int q(\mathbf{z}) \log p(\mathbf{x}) d\mathbf{z} \\
&= \underbrace{\int q(\mathbf{z}) \log q(\mathbf{z}) d\mathbf{z} - \int q(\mathbf{z}) \log p(\mathbf{x},\mathbf{z}) d\mathbf{z}}_{\text{negative ELBO}} + \log p(\mathbf{x}) \\
&= -\mathrm{ELBO} + \log p(\mathbf{x})
\end{aligned}
$$

Since the KL divergence is always non-negative, we have

$$
D_{\mathrm{KL}}\Big(q(\mathbf{z})\|p(\mathbf{z}|\mathbf{x})\Big) = -\mathrm{ELBO} + \log p(\mathbf{x}) \geq 0
$$

$$
\log p(\mathbf{x}) \geq \mathrm{ELBO}
$$

$\log p(\mathbf{x})$ is called the evidence, so ELBO gets its name (evidence lower bound). ELBO has different forms:

$$
\begin{aligned}
\mathrm{ELBO} &= -\int q(\mathbf{z}) \log q(\mathbf{z}) d\mathbf{z} + \int q(\mathbf{z}) \log p(\mathbf{x},\mathbf{z}) d\mathbf{z} \\
&= \int q(\mathbf{z}) \log \frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z})} d\mathbf{z} \\
&= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})} \log \frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z})}
\end{aligned}
$$

$$
\mathrm{ELBO} = \log p(\mathbf{x}) - D_{\mathrm{KL}}\Big(q(\mathbf{z})\|p(\mathbf{z}|\mathbf{x})\Big)
$$

For given observations, $\log p(\mathbf{x})$ is fixed. So to minimize $D_{\mathrm{KL}}\Big(q(\mathbf{z})\|p(\mathbf{z}|\mathbf{x})\Big)$ is to maximize the ELBO.

> KL Divergence
>
> Kullback-Leibler divergence (KL divergence) is a measure of the distance between two probability distributions. It is defined as:
>
> $$
> D_{\mathrm{KL}}\Big(q(\mathbf{x}\|p(\mathbf{x}))\Big) = \int q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p(\mathbf{x})} d\mathbf{x} = \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})} \log \frac{q(\mathbf{x})}{p(\mathbf{x})}
> $$
>
> The KL divergence is always non-negative, because

> $$\begin{aligned} 0 &= \log\left[\int p(\mathbf{x})d\mathbf{x}\right] \\ &= \log\left[\int q(\mathbf{x})\frac{p(\mathbf{x})}{q(\mathbf{x})}d\mathbf{x}\right] \\ &\geq \int q(\mathbf{x})\log\frac{p(\mathbf{x})}{q(\mathbf{x})}d\mathbf{x} \qquad \text{(Jensen's Inequality)} \end{aligned}$$
>
> Specifically, given $p=\mathcal{N}(\mu_1,\sigma_1^2)$, $q=\mathcal{N}(\mu_2,\sigma_2^2)$, we have:
>
> $$D_{\mathrm{KL}}(p\|q)=\ln\frac{\sigma_2}{\sigma_1}+\frac{\sigma_1^2+(\mu_1-\mu_2)^2}{2\sigma_2^2}-\frac{1}{2}$$
>
> Generally, in multivariate case, given $p=\mathcal{N}(\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1)$, $q=\mathcal{N}(\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2)$, we have:
>
> $$D_{\mathrm{KL}}(p\|q)=\frac{1}{2}\left[(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^T\boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)-\log\det(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1)+\mathrm{Tr}\left(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1\right)-n\right]$$
>
> Reference: 两个多元正态分布的KL散度、巴氏距离和W距离

> Jensen's Inequality
>
> If $\phi:\mathbb{R}\to\mathbb{R}$ is a real-valued convex function on $\Omega\subset\mathbb{R}^n$, then for any $f,g:\Omega\to\mathbb{R}$, we have:
>
> $$\phi\left[\int_\Omega f(\mathbf{x})g(\mathbf{x})d\mathbf{x}\right]\geq\int_\Omega\phi\left[f(\mathbf{x})\right]g(\mathbf{x})d\mathbf{x}$$
>
> For instance, assume $\phi(x)=\log(x)$, f,g are probability density functions on $\mathbb{R}^n$, we have
>
> $$\log\left[\int_\Omega f(\mathbf{x})g(\mathbf{x})d\mathbf{x}\right]\geq\int_\Omega\log\left[f(\mathbf{x})\right]g(\mathbf{x})d\mathbf{x}$$

VAE Model Architecture

Let us consider some dataset $\mathbf{X}=\{\mathbf{x}^{(i)}\}_{i=1}^N$ consisting of N i.i.d. samples of some continuous or discrete variable $\mathbf{x}\in\mathbb{R}^{d_1}$. We assume that the data are generated by some random process, involving an unobserved continuous random variable $\mathbf{z}\in\mathbb{R}^{d_2}$.

The generative model is given by the joint distribution of $\mathbf{x}$ and $\mathbf{z}$:

$$p_\theta(\mathbf{x},\mathbf{z})=p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})$$

1. A value $\mathbf{z}$ is generated from some prior distribution $p_\theta(\mathbf{z})$;
2. A value $\mathbf{x}$ is generated from some conditional distribution $p_\theta(\mathbf{x}|\mathbf{z})$.

To build the generative model, an idea is to introduce the posterior distribution of $\mathbf{z}$ given $\mathbf{x}$: $p_\theta(\mathbf{z}|\mathbf{x})$. The training process becomes:

1. Sample an observation $\mathbf{x}$;
2. Generate the latent variable $\mathbf{z}$ from the posterior distribution $p_\theta(\mathbf{z}|\mathbf{x})$;
3. Decode the latent variable $\mathbf{z}$ with $p_\theta(\mathbf{x}|\mathbf{z})$ get the reconstructed data $\mathbf{x}'$.
4. Update the parameters θ using some loss with $\mathbf{x}$ and $\mathbf{x}'$.

In practice, $p_\theta(\mathbf{z}|\mathbf{x})$ is intractable to compute, so we introduce $q_\phi(\mathbf{z}|\mathbf{x})$ to approximate it. The model can be viewed as an encoder-decoder architecture:

- Encoder: $q_\phi(\mathbf{z}|\mathbf{x})$
- Decoder: $p_\theta(\mathbf{x}|\mathbf{z})$

With maximum likelihood estimation, we write the objective as:

$$\begin{aligned} p_\theta(\mathbf{x}) &= \int p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z}) d\mathbf{z} \\ &= \int q_\phi(\mathbf{z}|\mathbf{x}) \frac{p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} d\mathbf{z} \\ &= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\frac{p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \right], \end{aligned}$$

Maximizing the log likelihood is equivalet to maximizing the ELBO:

$$\begin{aligned} \log p_\theta(\mathbf{x}) &= \log \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})} \left[\frac{p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \right] \\ &\geq \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})} \left[\log \frac{p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \right] \stackrel{\text{def}}{=} \text{ELBO}. \end{aligned}$$

Now we change the form of the ELBO:

$$\begin{aligned} \text{ELBO} &= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\log \frac{p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \right] \\ &= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\log \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \right] \\ &= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\log \frac{p_\theta(\mathbf{x}) p_\theta(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})} \right] \\ &= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}) + \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\log \frac{p_\theta(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})} \right] \\ &= \log p_\theta(\mathbf{x}) - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p(\mathbf{z}|\mathbf{x})} \right] \\ &= \log p_\theta(\mathbf{x}) - D_{\text{KL}} \Big(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}|\mathbf{x}) \Big), \end{aligned}$$

So when we maximize the ELBO, we are maximizing the evidence $\log p_\theta(\mathbf{x})$, as well as minimizing the KL divergence $D_{\text{KL}} \Big(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}|\mathbf{x}) \Big)$.

For interpretability, we can also rewrite the ELBO as:

$$\begin{aligned} \text{ELBO} &= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\log \frac{p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \right] \\ &= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) + \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\log \frac{p_\theta(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \right] \\ &= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z})} \right] \\ &= \underbrace{\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z})}_{\text{Reconstruction Error}} - \underbrace{D_{\text{KL}} \Big(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}) \Big)}_{\text{KL Loss}} \end{aligned}$$

So our objective only involves three distributions:

- $q_\phi(\mathbf{z}|\mathbf{x})$: Encoder, which is usually modeled as a neural network;
- $p_\theta(\mathbf{x}|\mathbf{z})$: Decoder, which is usually modeled as a neural network;
- $p_\theta(\mathbf{z})$: The prior distribution of the latent variable. We usually use some simple distribution with good properties, e.g., standard normal distribution.

By maximizing ELBO, we can learn the optimal parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ of the encoder and decoder.

VAE with Gaussian Prior

Now the problem becomes how to parameterize $q_\phi(\mathbf{z}|\mathbf{x})$, $p_\theta(\mathbf{z})$, and $p_\theta(\mathbf{z})$.

KL Loss

For the prior distribution of $\mathbf{z}$, we assume a standard normal distribution:

$$p_\theta(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$$

For the posterior distribution of $\mathbf{z}$, we assume a normal distribution

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\sigma}^2_\phi(\mathbf{x})\mathbf{I})$$

where $\boldsymbol{\mu}_\phi(\mathbf{x})$ and $\boldsymbol{\sigma}^2_\phi(\mathbf{x})$ are neural networks parameterized by ϕ.

Thanks to the good properties of normal distribution, we can compute $D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})\right)$ analytically. Since $p_\theta(\mathbf{z})$ and $q_\phi(\mathbf{z}|\mathbf{x})$ are both independent across dimensions, the KL divergence is simply the sum of the KL divergences of each dimension. For each dimension, we have:

$$\begin{aligned}
&D_{\mathrm{KL}}\left[\mathcal{N}(z;\mu,\sigma^2) \| \mathcal{N}(z;0,1)\right] \\
&= \int \mathcal{N}(z;\mu,\sigma^2) \log \frac{N(z;\mu,\sigma^2)}{N(z;0,1)} \mathrm{d}z \\
&= \int \mathcal{N}(z;\mu,\sigma^2) \log \frac{\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(z-\mu)^2}{2\sigma^2}\right)}{\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{z^2}{2}\right)} \mathrm{d}z \\
&= \int \mathcal{N}(z;\mu,\sigma^2) \left(-\log\sigma - \frac{(z-\mu)^2}{2\sigma^2} + \frac{z^2}{2}\right) \mathrm{d}z \\
&= \frac{1}{2}\int \mathcal{N}(z;\mu,\sigma^2) \left(-2\log\sigma - \frac{(z-\mu)^2}{\sigma^2} + z^2\right) \mathrm{d}z \\
&= \frac{1}{2}\left[-\log\sigma^2 \underbrace{\int \mathcal{N}(z;\mu,\sigma^2)\mathrm{d}z}_{1} - \frac{1}{\sigma^2}\underbrace{\int (z-\mu)^2\mathcal{N}(z;\mu,\sigma^2)\mathrm{d}z}_{\text{variance, } \sigma^2} + \underbrace{\int z^2\mathcal{N}(z;\mu,\sigma^2)\mathrm{d}z}_{\text{second moment, } \mu^2+\sigma^2}\right] \\
&= \frac{1}{2}\left(-\log\sigma^2 - \frac{1}{\sigma^2}\sigma^2 + \mu^2 + \sigma^2\right) \\
&= \frac{1}{2}\left(-1 - \log\sigma^2 + \mu^2 + \sigma^2\right)
\end{aligned}$$

Denote the dimension of $\mathbf{z}$ as d_2, the KL loss term can be expresses as:

$$D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})\right) = \sum_{i=1}^{d_2} \frac{1}{2}\left(-1 - \log\sigma_i^2 + \mu_i^2 + \sigma_i^2\right)$$

Reconstruction Error

We model the likelihood of $\mathbf{x}$ given $\mathbf{z}$ as a normal distribution:

$$\begin{aligned}
p_\theta(\mathbf{x}|\mathbf{z}) &= N(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I}) \\
&= \prod_{i=1}^{d_1} \mathcal{N}(x_i; \mu_i, \sigma_i^2) \\
&= \left(\prod_{i=1}^{d_1} \frac{1}{\sqrt{2\pi}\sigma_i}\right) \exp\left(-\sum_{i=1}^{d_1} \frac{(x_i-\mu_i)^2}{2\sigma_i^2}\right)
\end{aligned}$$

So

$$\begin{aligned}
\log p_\theta(\mathbf{x}|\mathbf{z}) &= \sum_{i=1}^{d_1} \log\left(\frac{1}{\sqrt{2\pi}\sigma_i}\right) - \sum_{i=1}^{d_1} \frac{(x_i-\mu_i)^2}{2\sigma_i^2} \\
&= -d_1 \log\sqrt{2\pi} - \frac{1}{2}\sum_{i=1}^{d_1}\log\sigma_i^2 - \frac{1}{2}\sum_{i=1}^{d_1}\frac{(x_i-\mu_i)^2}{\sigma_i^2}
\end{aligned}$$

We usually assume σ_i^2 are a same constant for all dimensions. So to maximize $\log p_\theta(\mathbf{x}|\mathbf{z})$ is to minimize $\sum_{i=1}^{d_1}(x_i-\mu_i)^2$.

The decoder network can learn to predict the mean of the distribution, i.e., $\boldsymbol{\mu} = (\mu_1, \mu_2, \cdots, \mu_{d_1})$, which is used as the reconstructed sample $\hat{\mathbf{x}}$. The reconstruction error becomes MSE loss between the original sample $\mathbf{x}$ and the reconstructed sample $\hat{\mathbf{x}}$.

Reparameterization Trick

The encoder of VAE doesn't give $\mathbf{z}$ directly. Instead, it gives $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ of $\mathbf{z}$'s distribution $q_\phi(\mathbf{z}|\mathbf{x})$. $\mathbf{z}$ is sampled from the distribution.

To use gradient descent to optimize the encoder network $q_\phi(\mathbf{z}|\mathbf{x})$, we need the gradient of $\mathbf{z}$ to $\boldsymbol{\phi}$:

$$\frac{\partial \mathbf{z}}{\partial \boldsymbol{\phi}} = \frac{\partial \mathbf{z}}{\partial \boldsymbol{\mu}} \frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\phi}} + \frac{\partial \mathbf{z}}{\partial \boldsymbol{\sigma}^2} \frac{\partial \boldsymbol{\sigma}^2}{\partial \boldsymbol{\phi}}$$

Previously, we have parameterized $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ as neural networks: $\boldsymbol{\mu} = \boldsymbol{\mu}_\phi(\mathbf{x}),\ \boldsymbol{\sigma}^2 = \boldsymbol{\sigma}^2_\phi(\mathbf{x})$. So $\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\phi}}$ and $\frac{\partial \boldsymbol{\sigma}^2}{\partial \boldsymbol{\phi}}$ can be computed. But $\frac{\partial \mathbf{z}}{\partial \boldsymbol{\mu}}$ and $\frac{\partial \mathbf{z}}{\partial \boldsymbol{\sigma}^2}$ cannot be directly computed, because $\mathbf{z}$ is sampled from $\mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I})$, and sampling isn't a differentiable operation.

To solve the problem, we can use the reparameterization trick. Sample a noise $\boldsymbol{\epsilon} \in \mathbb{R}^{d_2}$ from a standard normal distribution, $\mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{I})$, and we have

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon} \overset{\text{def}}{=} f(\boldsymbol{\epsilon}, \boldsymbol{\mu}, \boldsymbol{\sigma}^2) \overset{\text{def}}{=} g_{\boldsymbol{\epsilon}}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$$

$\boldsymbol{\epsilon}$ introduces randomness in generating $\mathbf{z}$ from $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$, but after it is sampled, the rule to generate $\mathbf{z}$ from $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ is fixed to a deterministic function $g_{\boldsymbol{\epsilon}}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$. So now we can compute $\frac{\partial \mathbf{z}}{\partial \boldsymbol{\mu}}$ and $\frac{\partial \mathbf{z}}{\partial \boldsymbol{\sigma}^2}$ using the chain rule as

$$\frac{\partial \mathbf{z}}{\partial \boldsymbol{\mu}} = \frac{\partial g_{\boldsymbol{\epsilon}}(\boldsymbol{\mu}, \boldsymbol{\sigma})}{\partial \boldsymbol{\mu}}, \quad \frac{\partial \mathbf{z}}{\partial \boldsymbol{\sigma}^2} = \frac{\partial g_{\boldsymbol{\epsilon}}(\boldsymbol{\mu}, \boldsymbol{\sigma})}{\partial \boldsymbol{\sigma}^2}$$

To summarize, we randomly sample $\boldsymbol{\epsilon}$ from $\mathcal{N}(\mathbf{0}, \mathbf{I})$ every time to determine a map $g_{\boldsymbol{\epsilon}}$, and generate $\mathbf{z} = g_{\boldsymbol{\epsilon}}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$, which is differentiable to $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$.