به نام خدا

دانشگاه صنعتی شریف

دانشکده مهندسی برق

# گزارش پروژه کارشناسی 1

عنوان :

# استنباط تغییرپذیر ، تعمیم های آن و کاربرد آن در Variational AutoEncoders

استاد پروژه : دکتر یاسایی

استاد درس : دکتر اکبر

نگارش :

حامد آجورلو

تابستان 1401

## فهرست مطالب

## 1. چکیده

یکی از مهم ترین مسائل در دنیای فعلی یادگیری ماشین این است که چگونه توزیع های آماری ای که به سادگی قابل محاسبه نیستند را تخمین بزنیم. به طور مثال در بسیاری از کاربردهای آمار بیزی، محاسبه ی توزیع پسین کار دشواری ست و به جای محاسبه ی دقیق سعی می شود تقریبی از آن بدست بیاید. برای این تقریبها به طور سنتی از روش های نمونه برداری مثل MCMCاستفاده می شده است. این روش های مبتنی بر نمونه برداری، وقتی ابعاد مسئله بالا می رود یا مدلها پیچیده می شوند ناکارآمد می شوند و قابل استفاده نیستند و استنباط تغییر پذیر جایگزین مناسبی برای آنهاست.

در استنباط تغییر پذیر به جای نمونه برداری برای تقریب توزیعها، سعی میکنیم به کمک یک بهینه سازی این کار را انجام دهیم. در این روش ابتدا خانواده ای از توزیع ها را در نظر میگیریم. سپس سعی میکنیم توزیعی در این خانواده را پیدا کنیم که کمترین انحراف را با توزیع مورد نظر ما داشته باشد. طبیعتا این جا نیاز است که معیاری برای انحراف دو توزیع در نظر بگیریم. در ابتدا برای محاسبه ی توزیع بهینه، از انحراف KL استفاده شده است ولی در پژوهش هایی که در سالهای اخیر انجام شده سعی شده است که این روش را به انحراف های دیگر نیز تعمیم بدهند. به طور مثال این روش به انحراف آلفا- بتا، انحراف هندسی جنسن - شانون و هر انحرافی f-تعمیم داده شده است که در برخی از مقالاتی که در انتها لینک آنها آورده شده، بررسی شده اند.

یکی از کاربردهای مهمی که استنباط تغییرپذیر در سالهای اخیر پیدا کرده است، استفاده از آن در مدل های مولد می باشد. در مدل های مولد سعی می شود با مشاهده ی یک مجموعه از داده هایی که از یک توزیع مشخص تولید شده اند، بتوان داده هایی بسیار شبیه به آنها که از همان توزیع می آیند را تولید کرد. به عنوان مثال فرض کنید هدف یک مدل مولد میتواند این باشد که با مشاهده ی تعداد زیادی عکس از چهره ی انسان بتواند، یک عکس جدید از یک چهره تولید کند. روش های مختلفی برای ساخت مدل های مولد وجود دارد و امروزه از مهم ترین این روش ها variational autoencoder ها یا به اختصار VAE ها می باشند. در این مدل فرض می شود که داده های مشاهده شده (مثلا تصاویر) بر اساس یک فضای متغیرهای پنهان تولید می شوند که معمولا بعد فضای متغیرهای پنهان کمتر از بعد خود داده است. یک VAE از دو بخش کدکننده و کدگشا تشکیل می شود که به صورت سری به هم متصل اند.

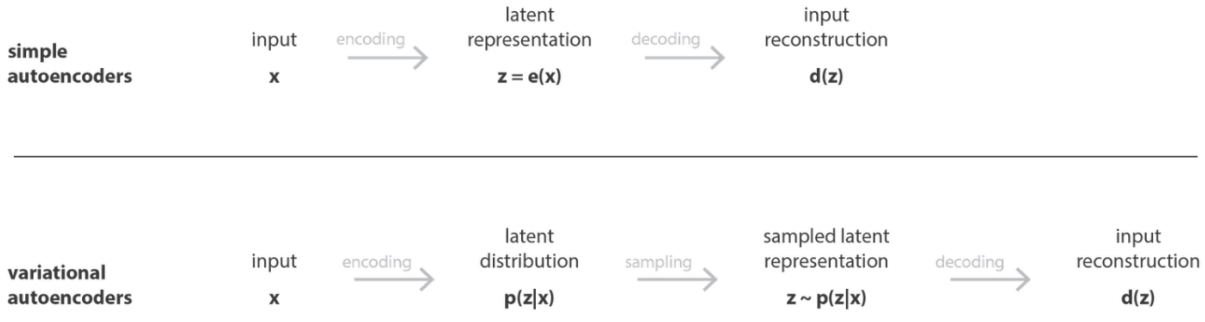



Figure 1 : *تفاوت بین simple autoencoder و variational autoencoder*

هدف یک autoencoder این است که با مشاهده ی داده , آن را توسط کدکننده به فضای متغیرهای پنهان ببرد و سپس از آن فضا توسط کدگشا یک داده ی تا جای ممکن شبیه به داده ی اصلی تولید کند. فرآیند یادگیری به این صورت است که کدکننده سعی میکند پارامترهای خود را طوری تنظیم کند که توزیع پسین روی متغیرهای پنهان به شرط مشاهده ی داده ها ماکزیمم شود و سپس کدگشا سعی می کند تا جای ممکن توزیع داده به شرط متغیرهای پنهان را بیابد و بر اساس آن داده ای مشابه با داده ی ورودی تولید کند. در نتیجه می توان گفت هدف کدکننده این خواهد بود که توزیع پسین روی متغیرهای پنهان به شرط مشاهده ی داده ها را تخمین بزند و در اینجا نیز به علت پیچیدگی محاسبه ی توزیع پسین روی متغیرهای پنهان، روش های تغییر پذیر خود را نشان می دهند. یعنی ابتدا خانواده ای از توزیع ها را در نظر میگیریم و سپس سعی میکنیم توزیعی در این خانواده را پیدا کنیم که کمترین انحراف را با توزیع مورد نظر ما داشته باشد. ما در پژوهش خود بخشی از تمرکز را بر روی مطالعه ی جنبه های مختلف و مرتبط VAE ها خواهیم گذاشت.

## 2. استنباط تغییر پذیر با انحراف KL

### 2.1 تعریف مسئله

همان طور که در مقدمه توضیح داده شد، پیدا کردن توزیع پسین در مسائل فعلی یادگیری ماشن جایگاه ویژه ای دارد. با دیدن هر داده ی جدید در استباط بیزی باید توزیع احتمال پارامترها را آپدیت کنیم.

$$p(z|x) = \frac{p(z,x)}{p(x)} = \frac{p(z)p(x|z)}{p(x)} \qquad , \qquad p(x) = \int p(z,x)dz$$

محاسبه ی توزیع پسین مخصوصا جمله ی p(x) کار ساده ای نیست . زیرا نیاز است روی همه ی متغیرهای نهان انتگرال بگیریم. در استنباط تغییر پذیر هدف اصلی این است که توزیع پسین در استنباط بیزی را با توزیعی ساده تر و قابل محاسبه تر تقریب بزنیم. به زبان ریاضی هدف پیدا کردن توزیع بهینه $q$ می باشد به طوری که:

$$q^*(z) = \ arg\min_{q(z)\in Q} D_{KL}(q(z)||p(z|x))$$

دقت کنید که برای اینکه محاسبه پذیری ساده تر شود توزیع q را اول می نویسیم تا امید ریاضی ها روی q باشد. حال با مقداری ساده سازی داریم:

$$D_{KL}(q(z)||p(z|x)) = E[\log q(z)] - E[\log p(z|x)]$$
$$= E[\log q(z)] - E[\log p(z,x)] + \log\ p(x)$$
$$= -ELBO + \log p(x)$$
$$= \ -\mathcal{L}(q) + \log p(x)$$

از آنجا که جمله ی log p(x) مستقل از توزیع q می باشد، واضح است که :

$$q^*(z) = \ arg\min_{q(z)\in Q} D_{KL}(q(z)\big|\big|p(z|x)\big) = \operatorname*{argmax}_{q(z)\in Q} \mathcal{L}(q)$$

همچنین اگر طرف چب و راست تساوی را جابه جا کنیم داریم:

$$\log p(x) = \ D_{KL}(q(z)||p(z|x)) + \ \mathcal{L}(q) \ \geq \ \mathcal{L}(q)$$

پس همانطور که واضح است $\mathscr{L}(q)$ یک کران پایین برای log p(x) می باشد. علتی که روی $\mathscr{L}(q)$ نام ELBO نیز می گذارند همین مورد است.

## 2.2 . الگوریتم های حل استنباط تغییر پذیر

### 2.2.1 افزایش مولفه

در این روش در هر گام تمامی مولفه های توزیع به جز یک مولفه را ثابت فرض می کنیم و در راستای آن مولفه بهینه سازی را انجام می دهیم. با تکرار این فرآیند به جواب بهینه خواهیم رسید.

برای اینکه بتوانیم این کار را انجام بدیم نیاز است فرض کنیم توزیع پسین مستقل است. به این فرض میدان میانگین می گویند. یعنی داریم:

$$q(z) = \prod_{j=1}^{m} q_j(z_j)$$

با این فرض اگر معادله را برای مولفه ی j ام مرتب کنیم داریم:

$$q^*(z) = \underset{q(z)\in Q}{\operatorname{argmax}}\, \mathcal{L}(q)$$

$$\mathcal{L}(q) = \int q(z_j)\, E_{-j}[\log p(z_j, z_{-j}, x)]dz_j - \int q(z_j) \log q(z_j) dz_j + c_j$$

$$\Rightarrow q_j^*(z_j) \propto \exp\{E_{-j}[\log p(z_j | z_{-j}, x)]\}$$

پس الگوریتم افزایش مولفه به شرح زیر است

**Algorithm 1:** Coordinate ascent variational inference (CAVI)

**Input**: A model $p(\mathbf{x}, \mathbf{z})$, a data set $\mathbf{x}$
**Output**: A variational density $q(\mathbf{z}) = \prod_{j=1}^{m} q_j(z_j)$
**Initialize:** Variational factors $q_j(z_j)$
**while** *the ELBO has not converged* **do**
  **for** $j \in \{1, \ldots, m\}$ **do**
    Set $q_j(z_j) \propto \exp\{\mathbb{E}_{-j}[\log p(z_j \mid \mathbf{z}_{-j}, \mathbf{x})]\}$
  **end**
  Compute $\text{ELBO}(q) = \mathbb{E}[\log p(\mathbf{z}, \mathbf{x})] + \mathbb{E}[\log q(\mathbf{z})]$
**end**
**return** $q(\mathbf{z})$

*Figure 2 : الگوریتم افزایش مرتبه*

در الگوریتم افزایش مولفه نیاز است که ابتدا بتوانیم به طور تحلیلی مقدار امید ریاضی داده شده را حساب کنیم که در خیلی مواقع مانند شبکه های یادگیری عمیق ممکن نیست و همچنین برای محاسبه پذیری نیاز به استفاده از خانواده همزاد داریم که دست ما را می بندد.

### 2.2.2 روش افزایش گرادیان

ابتدا فرض می کنیم که خانواده توزیع $Q$ با پارامتر $\lambda$ پارامتری شده باشد. برای بدست آوردن لاندای بهینه از روش افزایش

گرادیان روی تابع استفاده میکنیم یعنی داریم:

$$\lambda_{t+1} = \lambda_t + \gamma_t \nabla_\lambda \mathcal{L}(\lambda_t)$$

مقدار گرادیان تابع $\mathcal{L}$ به صورت زیر بدست می آید.

$$\nabla_\lambda \mathcal{L} = E_q[\nabla_\lambda \log q(z;\lambda)(\log p(x,z) - \log q(z;\lambda))]$$

واضح است محاسبه ی گرادیان بالا نیاز به امید ریاضی گرفتن روی توزیع q دارد که مسئله ی سختی ست. برای بر طرف شدن این مشکل از دو ایده ی مختلف که در زیر آورده شده است استفاده می کنیم. هر دوی این روش ها از افزایش گرادیان تصادفی برای بدست آوردن جواب استفاده میکنند و فقط تخمین آنها از بردار گرادیان متفاوت می باشد.

#### REINFORCE Gradients

مقدار گرادیان را به کمک سمپل گرفتن از توزیع q (که چون توزیع نسبتا ساده ای هست می توان از آن سمپل گرفت) تخمین می زنیم و داریم:

$$\nabla_\lambda \hat{\mathcal{L}} = \frac{1}{K}\sum_{k=1}^{K} \nabla_\lambda \log q(z_k;\lambda)(\log\ p(x,z_k) - \log q(z_k;\lambda))$$

در استفاده های مختلف دیده شده است که تخمین گر بالا باعث به وجود آوردن واریانس زیاد در $\lambda$ می شود که برای کاهش آن از دو روش Rao-Blackwellization و Control variates استفاده می شود که برای طولانی نشدن گزارش از توضیح آنها صرف نظر میکنیم.

#### Reparameterization Gradient

در این روش که به طور تجربی مشاهده شده است که دارای واریانس کمتری از روش قبلی می باشد، متغیر تصادفی z را بر حسب یک تابع قطعی از نویز و $\lambda$ می نویسیم. یعنی داریم:

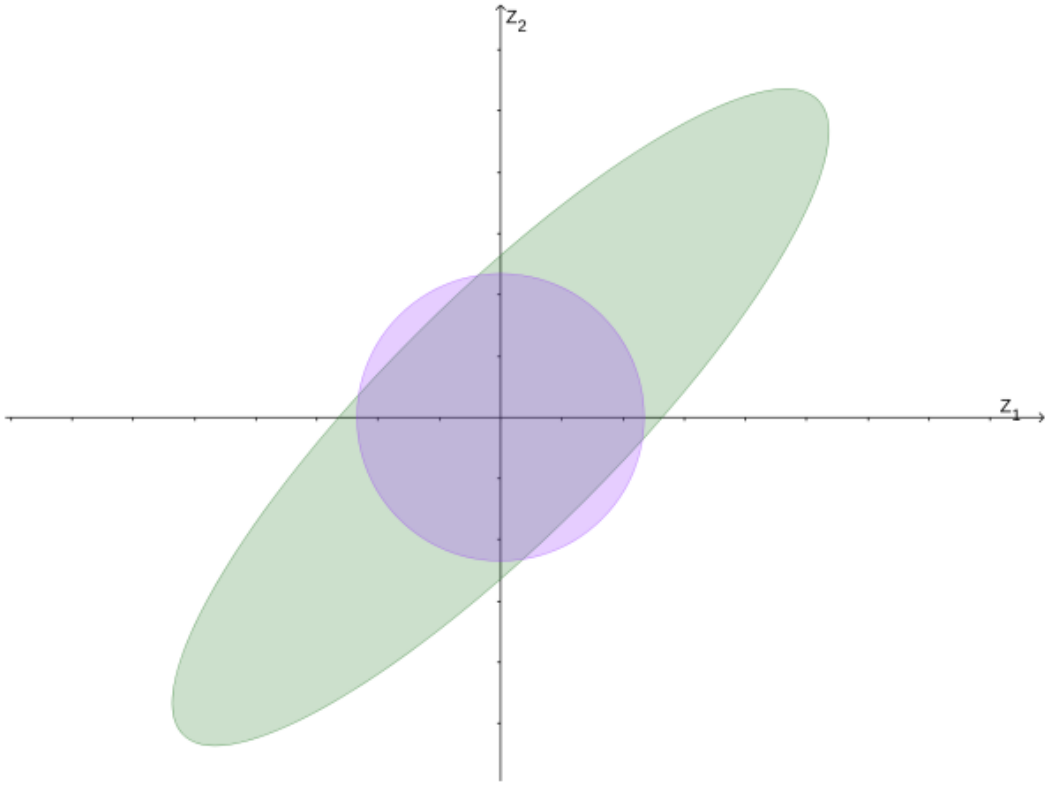
$$z \sim q(z;\lambda) \Rightarrow z = g(\epsilon,\lambda)\ , \epsilon \sim r(\epsilon)$$

سپس با سمپل گرفتن از نویز داریم:

$$\nabla_\lambda \hat{\mathcal{L}}_{rep} = \frac{1}{K}\sum_{k=1}^{K} \nabla_\lambda \log q(g(\epsilon_k,\lambda);\lambda)(\log\ p(x,g(\epsilon_k,\lambda)) - \log q(g(\epsilon_k,\lambda);\lambda))$$

## 3.2 محدودیت های روش استنباط تغییر پذیر با انحراف KL

در تخمین به کمک KL واریانس تخمینی ما کمتر از واریانس واقعی می باشد زیرا با توجه به تعریف انحراف KL هر جا که توزیع p دامنه ای ندارد توزیع q نیز ندارد. مثلا برای حالت فرض میدان میانگین به شکل زیر توجه کنید.




بیضی سبز نشان دهنده ی قسمت اصلی توزیع پسین و بیضی بنفش نشان دهنده ی قسمت اصلی توزیع q می باشد. توزیع q فقط در راستای محورها می تواند کشیده شود و بیرون از بیضی سبز هم نمی تواند برد پس واریانس کمتری خواهد داشت.

این کاهش واریانس باعث از بین رفتن مدهای مهم توزیع پسین می شود که به شدت نامطلوب است.

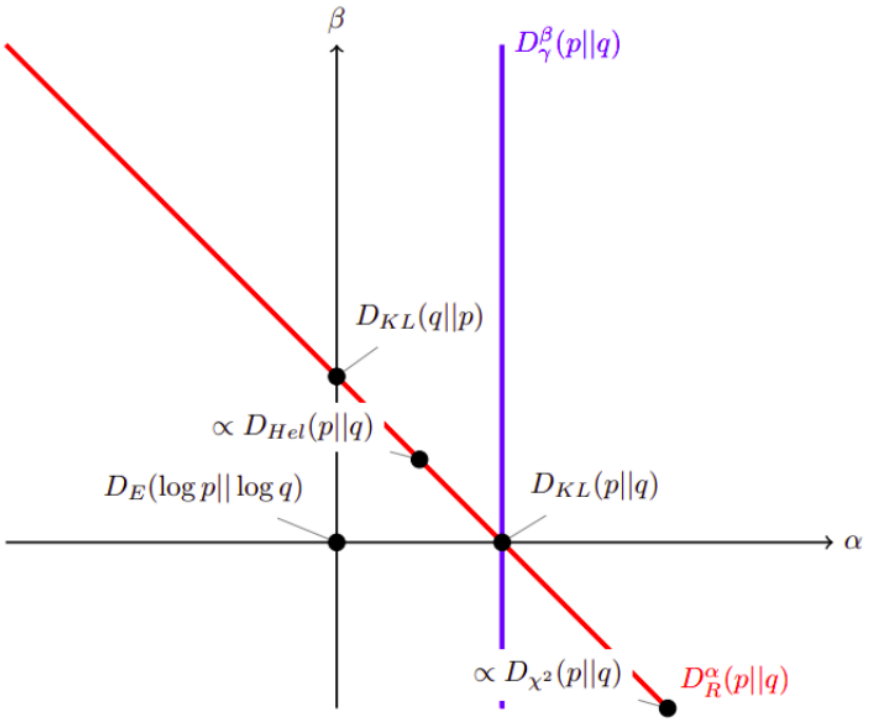
با کمک گرفتن از انحراف های دیگر می توان سعی بر رفع مشکل کاهش واریانس داشت. همچنین می توان از انحراف های دیگر به امید رسیدن به تخمین های بهتر استفاده کرد.

در سال های اخیر تحقیقاتی زیادی روی این حوزه شده که در فصل بعدی چند مورد از آنها را بررسی می کنیم.

# 3. استنباط تغییر پذیر در انحراف های دیگر

## 1.3 تلاش های انجام شده

مقالات مختلفی برای تعمیم استنباط وردشی در سال های اخیر وجود دارد که بتوانند به خاصیت هایی که برای تخمین وزیع پسین مورد پسند می باشد دست پیدا کنیم. مثلا در [7] سعی شده است به تخمینی برسیم که علاوه بر اینکه تمام جرم احتمال را می پوشاند در مقابل داده های پرت نیز مقاوم باشد. دو پارامتر آلفا و بتا در این انجراف وجود دارد که می توان با تعیین آنها به سمتی که مطلوب است حرکت کنیم. برای فهم بهتر این نکته به شکل زیر توجه کنید.




تلاش های متنوع دیگری نیز در این مورد انجام شده است که از ذکر آنها صرف نظر میکنیم.

## 2.3 استنباط تغییر پذیر برای f انحراف

در این جا به سراغ [۳] می رویم که سعی کرده تلاش های موجود در این حوزه را در تئوری های خود یکپارچه کند. در ابتدا نیاز به چند تعریف تعریف داریم. در ابتدا تابع دوگان به صورت زیر تعریف می شود

$$f^*(t) = t.f(\frac{1}{t})$$

در مورد f انحراف ها می دانیم :

$$D_{f^*}(p||q) = D_f(q||p)$$

در مقالات قبل از این مقاله نشان داده شده است که به طور کلی نمی توان مسئله استنباط تغییر پذیر را برای هرf انحرافی حل کرد به همین دلیل تمرکز این مقاله به روی انحراف هایی می باشد که تابع مولدی به شکل زیر دارند.

$$f_\lambda(.) = f(\lambda.) - f(\lambda)$$

توضیحاتی در مقاله در باب این که چرا این دسته از انحراف ها خوب هستند داده است که از آن می گذریم. برای این دسته از توابع می توان به سادگی دید که

$$D_{f_{p(D)^{-1}}}(q(z)||p(z|D)) = \frac{1}{p(D)}.E_{q(z)}\left[f^*\left(\frac{p(z,D)}{q(z)}\right)\right] - f(\frac{1}{p(D)})$$

و سپس با ساده کردن به تساوی زیر رسید.

$$\mathcal{L}_f(q,D) = E_{q(z)}\left[f^*\left(\frac{p(z,D)}{q(z)}\right)\right] = f^*\big(p(D)\big) + p(D).D_{p(D)^{-1}}(q(z)||p(z|D))$$

پس داریم :

$$\mathcal{L}_f(q,D) = E_{q(z)}\left[f^*\left(\frac{p(z,D)}{q(z)}\right)\right] \geq f^*\big(p(D)\big)$$

کران بدست آمده خیلی از کرانهای مقاله های قبلی رو به ما می دهد که در فایل پیوست مقاله مثال های زیادی داده شده است. مثلا اگر $f(t) = tlog$را در نظر بگیریم داریم $f^*(t) = tlog$ و به همان کران ELBO می رسیم. برای اینکه الگوریتم افزایش مولفه با فرض میدان میانگین را بتوان برای هر انحرافی اجرا کرد نیاز به تعریف دو خانواده ی زیر داریم:

$$f \in F_0 \Leftrightarrow f(t\hat{t}) = t^\gamma f(\hat{t}) + f(t)$$
$$f \in F_1 \Leftrightarrow f(t\hat{t}) = t^\gamma f(\hat{t}) + f(t)\hat{t}$$

حال اگر $f \in F_0$ داریم:

$$q_j^*(z_j) \propto f^{-1}(E_{q_{-j}}\left[f\left(\frac{p(z,D)}{q_{-j}(z_{-j})}\right)\right])$$

و اگر $f \in F_1$ داریم:

$$q_j^*(z_j) \propto f^{*-1}(E_{q_{-j}}\left[f^*\left(\frac{p(z,D)}{q_{-j}(z_{-j})}\right)\right])$$

تعداد زیادی ازf انحراف های معروف عضو یکی از خانواده های $F_0$ و یا $F_1$ می باشند. مثلا KL عضو $F_1$ است و$\chi^2$ عضو $F_0$ می باشد . در الگوریتم افزایش گرادیان نیاز به محاسبه ی تخمینی از گرادیان داریم. مقدار گرادیان برابر است با:

$$\nabla_\theta \mathcal{L}_f(q_\theta, D) = E_{q_{\theta(z)}}[f'\left(\frac{q_\theta(z)}{p(z,D)}\right).\nabla_\theta \log q_\theta(z)]$$

پس برای الگوریتم افزایش گرادیان تصادفی در حالت REINFORCE Gradients از تخمین زیر برای گرادیان استفاده میکنیم که با نمونه برداری از توزیع q بدست می آید:

$$\nabla_\theta \hat{\mathcal{L}}_f(q_\theta, D) = \frac{1}{K}\sum_{k=1}^{K} f'\left(\frac{q_\theta(z)}{p(z,D)}\right).\nabla_\theta \log q_\theta(z)$$

همانطور که قبلا هم اشاره شده بود نیاز است واریانس این تخمین گر را با روش هایی کاهش دهیم. در حالت Reparameterization Gradient به تخمین گر زیر برای گرادیان می رسیم:

$$\nabla_\theta \hat{\mathcal{L}}^{rep}{}_f(q_\theta, D) = \frac{1}{K}\sum_{k=1}^{K} \nabla_\theta f^*\left(\frac{p(g_\theta(\epsilon_k), D)}{q_\theta(g_\theta(\epsilon_k))}\right)$$

همانطوری که واضح است تخمین گرهای بالا به کل ورودی وابسته اند که در عمل کار را کند می کنند. برای رفع این مشکل نیز راهکارهایی پیشنهاد شده است که از شرح آنها صرف نظر میکنیم

# 4. Variational Auto Encoders

## 1.4 طرح مساله

در این قسمت می خواهیم یک مدل مولد (که در مقدمه توضیح داده شد) را بر اساس روش های تغییر پذیر که تا به حال آموخته ایم طراحی کنیم. ابتدا مساله را شرح می دهیم. در این مسائل ما فرض می کنیم مجموعه داده ی $X = \{x^{(i)}\}_{i=1}^{N}$ که نمونه های i.i.d از یک متغیر تصادفی مانند x هستند. فرض اصلی ما این است که داده بر اساس یک روش تصادفی معینی تولید می شود و همین فرض اساسی است که ما را در طراحی VAE ها یاری می کند. ما فرض می کنیم یک متغیر نهانی در مساله وجود دارد که داده ها بر اساس آن تولید می شوند. مثلا فرض کنید داده ها عکس باشند و این متغیر نهان به تعبیری فشرده ی اطلاعات هر عکس را در خود دارد. پس ما فرض می کنیم متغیر تصادفی پنهان $z^{(i)}$ از یک توزیع پارامتری $p_{\theta^*}(z)$ که توزیع پیشین روی فضای متغیرهای نهان ما است تولید می شود و سپس داده از روی توزیع شرطی (مدل مولد) $p_{\theta^*}(x|z)$ تولید می شود .

در اینجا دقت داریم که $\theta^*$ پارامترهای مدل مولد را توصیف می کنند که در عمل می توانند پارامترهای یک شبکه ی عصبی باشند. متاسفانه از این فرآیند هم پارامتر $\theta^*$ و هم متغیرهای نهان $z^{(i)}$ از ما پنهان هستند و ما فقط به $X = \{x^{(i)}\}_{i=1}^{N}$ دسترسی داریم. مشابه آنچه که قبل تر بیان شد در اینجا نیز توزیع پیشین که روی متغیرهای نهان بعد از مشاهده ی داده ها می افتد محاسبه پذیر نیست و ما با استفاده از روش های تغییر پذیر آن را تقریب خواهیم زد. پس در ادامه ما مدل تشخیص دهنده ی $q_\phi(z|x)$ را به عنوان تقریبی از توزیع پسین پیچیده ی واقعی به کار خواهیم برد. حال بر خلاف روش هایی که در استنباط تغییر پذیر معمولی به کار می رود (مانند تقریبmean field) . ما فرض نمیکنیم توزیع $q_\phi(z|x)$ و در راستاهای مختلف از هم مستقل باشد یا حتما امید ریاضی آن فرم بسته ای داشته باشد. به جای چنین فرض هایی ما ما یک روش یادگیری ارائه می دهیم که پارامترهای مدل تشخیص دهنده یعنی $\phi$ و پارامترهای مدل مولد یعنی $\theta$ را همزمان با هم یاد بگیرد. یک شمای کلی از ایده های مطرح شده را می توان در شکل زیر مشاهده کرد.

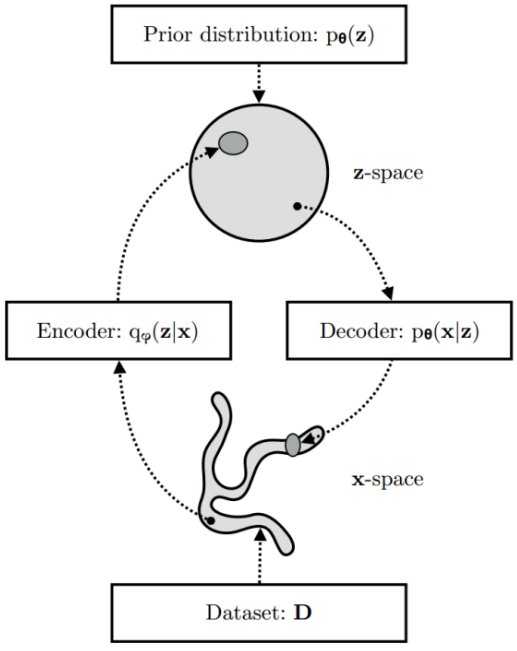



*Figure 3 : یک شمای کلی از طرحی که در VAE ها به کار می رود*

## 2.4 مدل یادگیری

در اینجا ما سعی میکنیم پارامترهای خود را طوری تعیین کنیم که احتمال مشاهده ی داده بیشترین مقدار خود را بیابد. پس سعی میکنیم مقدار پارامترها را طوری تنظیم کنیم که $\log p_\theta\left(x^{(1)}, x^{(2)}, \dots, x^{(N)}\right) = \sum_{i=1}^{N} log\ p_{\theta^*}(x^{(i)})$ ماکزیمم شود اما همان طور که بحث شده است این مقدار محاسبه پذیر نیست و به جای ماکزیمم کردن آن یک کران پایین روی آن را ماکزیمم می کنیم. این کران پایین همان ELBO مذکور خواهد بود.

$$\log p_\theta\left(x^{(i)}\right) = D_{KL}(q_\phi(z|x^{(i)})|\left|p_\theta\left(z|x^{(i)}\right)\right) + ELBO(\theta, \phi; x^{(i)})$$

که در آن ELBO به صورت زیر است:

$$ELBO\left(\theta, \phi; x^{(i)}\right) = E_{q_\phi(z|x)}[-\log q_\phi(z|x) + \log p_\theta(x, z)] \qquad (1.4)$$

$$= -D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z)) + E_{q_\phi\left(z|x^{(i)}\right)}[\log p_\theta(x^{(i)}|z)] \qquad (2.4)$$

که بیشینه کردن آن به معنای کاهش $D_{KL}\left(q_\phi\left(z|x^{(i)}\right)||p_\theta\left(z|x^{(i)}\right)\right) + ELBO(\theta, \phi; x^{(i)})$ و افزایش $E_{q_\phi(z|x^{(i)}}[\log p_\theta(x^{(i)}|z)$ است . کاهش ترم اول به معنای دور نشدن بیش از حد توزیع پسین تقریبی از توزیع پیشین است که از overfitجلوگیری میکند و نقش یک ترم regularization را بازی می کند. افزایش ترم دوم نیز بع معنای افزایش likelihood داده ها تحت توزیع $q_\phi\left(z|x^{(i)}\right)$ است. برای یافتن نقطه ی بهینه ی این کران پایین می توان از روش های گرادیان افزایشی بهره جست اما محاسبه ی این گرادیان نسبت به پارامتر $\phi$ سخت است. حتی روش های معمولی Monto Carlo برای تخمین گرادیان به صورت زیر هستند که نشان داده شده است دارای واریانس بسیار زیادی می باشند و در عمل به کار نمی آیند.

$$\nabla_\phi E_{q_\phi}[f(z)] = E_{q_\phi(z)}[f(z)\nabla_{q_\phi(z)} \log q_\phi(z) \cong \frac{1}{L}\sum_{l=1}^{L} f(z)\nabla_{q_\phi(z^{(l)})} \log q_\phi(z^{(l)})$$

پس نیاز است که به روش های دیگری ازz نمونه بگیریم. یک دلیل دیگر نیز که به روش های هوشمندانه تری برای نمونه گیری نیاز داریم این است که در ادامه می بینیم در VAE ها مدل های تشخیص دهند و مولد، شبکه های عصبی هستند و هنگامی که ما از روش های گرایان تصادفی برای بهینه کردن وزن آنها استفاده می کنیم، از روش backpropagation برای انتشار گرادیان از انتهای شبکه به ابتدای شبکه بهره می بریم. در حالتی که ما مستقیما از خود z نمونه گرفته باشیم، نمی توانیم انتشار گرادیان را از شبکه ی کدگشا ( مولد ) به شبکه ی کدکننده ( تشخیص دهنده) منتقل کنیم. در اثر همه ی این مشکلات به روشی به نام بازپرمایش پارامترها می رسیم.

## 3.4 بازپرمایش پارامترها

تحت یک سری مفروضات نه چندان سختگیرانه، برای یک توزیع پسین مانند $q_\phi(z|x)$ می توان متغیر تصادفی $\tilde{z} \sim q_\phi(z|x)$ را به کمک تابع مشتق پذیر $g_\phi(\epsilon, x)$ از یک نویز کمکی مانند $\varepsilon$ بازپرمایش پارامتری کرد.

$$\tilde{z} = g_\phi(\epsilon, x) \quad , \quad \epsilon \sim p(\epsilon)$$

به عنوان مثال فرض کنید $\tilde{z} \sim q_\phi(z|x) = \mathbb{N}(\mu_x, \sigma_x^2)$ است. می توان $\tilde{z}$ را به صورت $\tilde{z} = \mu_x + \sigma_x\epsilon$ نوشت که $\epsilon \sim \mathbb{N}(0,1)$ است. در نتیجه حالا می توان تخمین monte carlo از امید ریاضی یک تابع مانند f(z) که امید ریاضی نسبت به $q_\phi(z|x)$ گرفته می شود را به صورت زیر تشکیل داد.

$$E_{q_\phi(z|x^{(i)})}[f(z)] = E_{p(\varepsilon)}\left[f\left(g_\phi(\varepsilon, x^{(i)})\right)\right] \cong \frac{1}{L}\sum_{l=1}^{L} f\left(g_\phi(\epsilon^{(l)}, x^{(i)})\right) \quad , \quad \epsilon^{(l)} \sim p(\epsilon)$$

همچنین بدون مشکل می توان از روش backpropagation در آموزش شبکه های عصبی هنگامی که به عنوان مدل های تشخیص دهنده و مولد به کار می روند استفاده کرد.

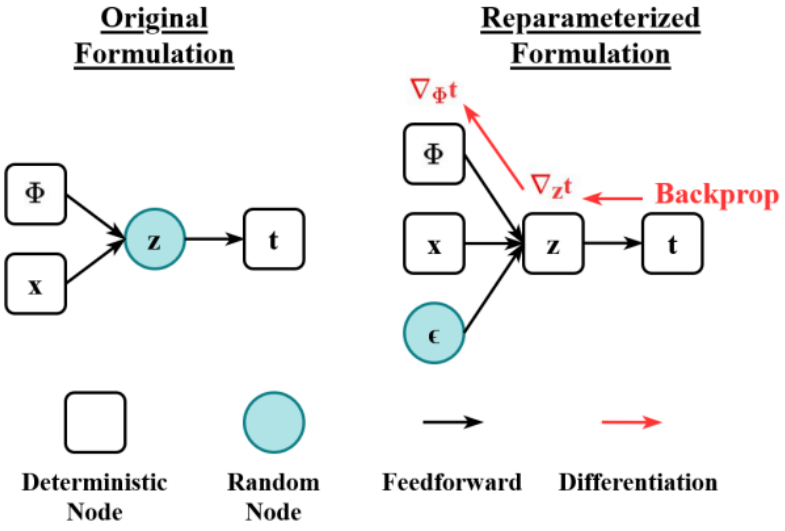



*Figure 4: اثر باز پرمایش پارامترها در آموزش شبکه های عصبی*

این روش بازپرمایش پارامترها را برای کران پایین وردشی خود یعنی ELBO نیز می توان به کار برد و می توان به الگوریتم کلی زیر برای یافتن پارامترهای مدل های مولد و تشخیص دهنده رسید.

**Algorithm 1** Minibatch version of the Auto-Encoding VB (AEVB) algorithm. Either of the two SGVB estimators in section 2.3 can be used. We use settings $M = 100$ and $L = 1$ in experiments.

$\theta, \phi \leftarrow$ Initialize parameters
**repeat**
  $\mathbf{X}^M \leftarrow$ Random minibatch of $M$ datapoints (drawn from full dataset)
  $\epsilon \leftarrow$ Random samples from noise distribution $p(\epsilon)$
  $\mathbf{g} \leftarrow \nabla_{\theta,\phi}\widetilde{\mathcal{L}}^M(\theta, \phi; \mathbf{X}^M, \epsilon)$ (Gradients of minibatch estimator (8))
  $\theta, \phi \leftarrow$ Update parameters using gradients $\mathbf{g}$ (e.g. SGD or Adagrad [DHS10])
**until** convergence of parameters $(\theta, \phi)$
**return** $\theta, \phi$

*Figure 5: الگوریتم AEVB برای یافتن پارامتر های مدل های مولد و تشخیص دهنده*

در الگوریتم فوق $\tilde{\mathcal{L}}^M$ همان تخمین ما از $\text{ELBO}(\theta, \phi; X)$ می باشد . به صورت دقیق تر داریم.

$$ELBO(\theta, \phi; X^M) \cong \hat{\mathcal{L}}^M(\theta, \phi; X^M) = \frac{N}{M}\sum_{i=1}^{M} \tilde{\mathcal{L}}\left(\theta, \phi; x^{(i)}\right) \qquad (3.4)$$

که در آن $\tilde{\mathcal{L}}(\theta, \phi; x^{(i)})$ تخمین ما از ELBO در یک نقطه می باشد. برای این کار طبق فرمولهای 1.4 یا 2.4 می توان بر اساس بازپرمایش پارامتری تخمینی برای ELBO به دست آورد. فرمول 1.4 از جهتی با فرمول2.4 متفاوت است. در فرمول 2.4 عبارت $D_{KL}(q_\phi\left(z|x^{(i)}\right)||p_\theta(z))$ ظاهر شده است که برای برخی از توزیع های معروف مانند توزیع نرمال که اتفاقا ما از آنها در VAE استفاده می کنیم، می توان آن را به صورت تحلیلی حساب کرد و در نتیجه نیازی به تخمین آن نمی باشد. در هر حال اگر از فرمول 1.4 برای تخمین زدن ELBO با بازپرمایش پارامترها استفاده کنیم، به تخمین گر زیر می رسیم

$$\tilde{\mathcal{L}}^A\left(\theta, \phi; x^{(i)}\right) = \frac{1}{L}\sum_{l=1}^{L} \log p_\theta\left(x^{(i)}, z^{(i,l)}\right) - \log q_\phi\left(z^{(i,l)}|x^{(i)}\right)$$

$$where \quad z^{(i,l)} = g_\phi\left(\epsilon^{(i,l)}, x^{(i)}\right) \quad , \quad \epsilon^{(l)} \sim p(\epsilon)$$

و اگر از 2.4 برای تخمین زدن ELBO با بازپرمایش پارامترها استفاده کنیم , به تخمینگر زیر می رسیم.

$$\tilde{\mathcal{L}}^B\left(\theta, \phi; x^{(i)}\right) = -D_{KL}(q_\phi\left(z|x^{(i)}\right)|\,|p_\theta(z)) + \frac{1}{L}\sum_{l=1}^{L}(\log p_\theta(x^{(i)}|z^{(i,l)}))$$

$$where \quad z^{(i,l)} = g_\phi\left(\epsilon^{(i,l)}, x^{(i)}\right) \quad , \quad \epsilon^{(l)} \sim p(\epsilon)$$

هر کدام از این دو تخمین گر را (که البته دومی به دلیل اینکه یک جمله آن به صورت تحلیلی قابل محاسبه است ارجحیت دارد) می توان در فرمول 3.4 قرار داد و الگوریتم Auto-Encoding Variational Bayes یا AEVB را با آن اجرا کرد.

به عنوان نکته نهایی لازم به ذکر است که اگر مدلهای کدگشا و کدکننده ( مولد و تشخیص دهنده) شبکه ی عصبی باشند به جای الگوریتم AEVBمی توان از ترکیب روش هایی که مخصوص آموزش شبکه های عصبی است با الگوریتم AEVB استفاده کرد که همان طور که ذکر شد بازپرمایش پارامترها کمک می کند چنین روش هایی قابل اجرا باشند.

## 4.4 مدل نهایی برای VAE

مدل کلی ارائه شده در قسمت های قبل را می توان کمی خاص کرد و به ساختار نهایی VAE رسید. در این قسمت ما برای کدکننده ی احتمالاتی خود یعنی $q_\phi(z|x)$ (که تقریب توزیع پسین از مدل مولد $p_\theta(z)p_\theta(x|z)$ می باشد.) از یک شبکه ی عصبی استفاده می کنیم. همچنین فرض می کنیم توزیع پیشین روی فضای پارامترهای پنهان یک بردار تصادفی نرمال استاندار باشد

که البته در آن وابستگی به پارامترهای مدل مولد وجود ندارد. همچنین فرض میکنیم مدل مولد ما یعنی $p_\theta(x|z)$ دارای توزیع نرمال (در صورت حقیقی مقدار بودن داده و یا توزیع برنولی (در صورت باینری بودن داده است بدین ترتیب که پارامتر نهان از یک شبکهی عصبی با پارامتر $\theta$ عبور میکند و توزیعی روی داده به وجود می آورد. حال با اینکه محدودیتی در انتخاب $q_\phi(z|x)$ نداریم فرض می کنیم توزیع پسین واقعی (البته غیر قابل محاسبه) نزدیک به یک توزیع گوسی با ماتریس کوواریانس قطری باشد و در نتیجه اجازه می دهیم توزیع وردشی تقریبی ما یک بردار گوسی با ماتریس کوواریانس قطری باشد.

$$log\ q_\phi(z|x^{(i)}) = \log \mathbb{N}(z; \mu^{(i)}, \sigma^{2(i)} I)$$

که بردار میانگین و واریانس ها در هر راستا از روی خروجی شبکه ی کدکننده (تشخیص دهنده) بر اساس داده ورودی تعیین می شوند. حال همان طور که در بخش قبل بیان شد، ما می توانیم از توزیع پسین تقریبی خود به صورت $z^{(i,l)} = g_\phi(x^{(i)}, \epsilon^{(l)}) = \mu^{(i)} + \sigma^{(i)} \odot$ که $\epsilon^{(l)} \sim \mathbb{N}(0, I)$ و $\odot$ نماد ضرب عنصر به عنصر است. در این حالت چون هم توزیع پسین تقریبی و هم توزیع پیشین نرمال است و انحراف KL بین آنها فاصله معنادار دارد می توانیم از تخمینگری که در قسمت قبل تحت عنوان $\tilde{\mathcal{L}}^{\mathcal{B}}$ بیان شد استفاده کنیم. در نهایت تخمین گری که در این مدل و برای داده ی $x^{(i)}$ به دست می آید به صورت زیر است. (که ما به دنبال بیشینه کردن آن هستیم)

$$\mathcal{L}(\theta, \phi; x^{(i)}) \cong \frac{1}{2} \sum_{j=1}^{J} \left(1 + \log\left(\left(\sigma_j^{(i)}\right)^2\right) - \left(\mu_j^{(i)}\right)^2 - \left(\sigma_j^2\right)^2\right) + \frac{1}{L} \sum_{l=1}^{L} \log p_\theta(x^{(i)}|z^{(i,l)})$$

$$where \quad z^{(i,l)} = \mu^{(i)} + \sigma^{(i)} \odot \epsilon^{(l)} \quad , \quad \epsilon^{(l)} \sim \mathbb{N}(0, I)$$

می توان تصویر کلی ای از VAE به صورت زیر در ذهن داشت.

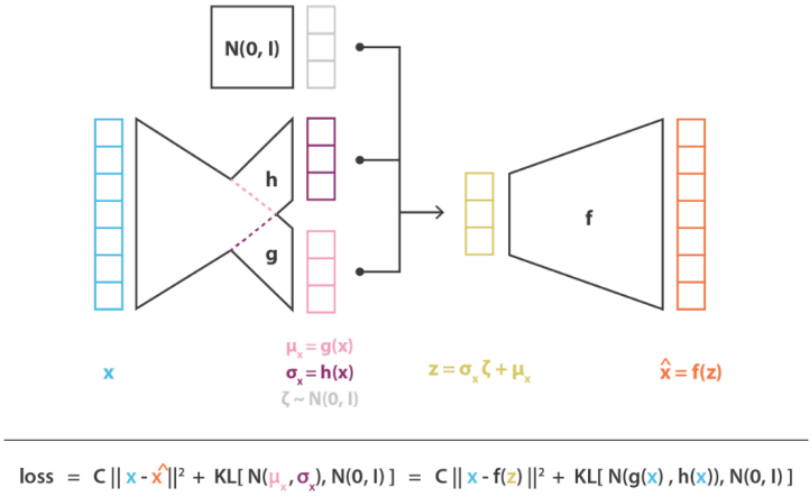



*Figure 6 : variational autoencoder یک توصیف کلی از یک*

هنگامی که شما از این ساختار بخواهید به عنوان مدل مولد استفاده کنید می توانید از متغیر پنهان بر اساس توزیع پشین با توزیع دیگری سمپل بگیرید و آن را از شبکه ی کدگشا عبور داد تا به یک داده ی جدید (مثلا یک تصویر از انسان) رسید.

## 5.4 توسعه ها و ایده های جدید در مدلVAE

پس از مطرح شدن ایدهی VAE ها، مقالات بسیاری به گسترش ایده های مختلف در ادامه این مسیر شدند. برخی از این مسیرها که به مطالب درس بیشتر مربوط بود را می توان به صورت زیر دسته بندی کرد:

1. استفاده از f-Divergence های دیگر به جای $D_{KL}$ در کران پایین تغییر پذیر.
2. اضافه کردن یک سری جملات دیگر به ترم Regularization در آموزش VAE ها.
3. استفاده از برخی روش های تکمیلی دیگر در آموزش VAE ها.

### 1.5.4 استفاده از f-Divergence های دیگر به جای $D_{KL}$ در کران پایین تغییر پذیر

در قسمت های قبلی به برخی مشکلاتی که $D_{KL}$ به عنوان متری که می خواهیم فاصله ی توزیع پسین تقریبی خود را با توزیع پسین واقعی کم کنیم، بیان کردیم. بسیاری از مقالات سعی کردند از توابع دیگری به جای $D_{KL}$ در کم کردن این فاصله استفاده کنند. یکی از خوبی هایی که $D_{KL}$ دارد محاسبه پذیری آن برای توزیع های نرمال است. در نتیجه حالت مطلوب برای ما یافتن انحرافی است که مشکلات $D_{KL}$ را نداشته باشد ولی حسن آن که محاسبه پذیری روی توزیع های نرمال است را داشته باشد. یکی از انحراف هایی که چنین خاصیت خوبی را برآورده می کند انحراف هندسیJensen-Shannon می باشد. این انحراف به صورت زیر تعریف می شود

$$JS^{G_\alpha}(p(x)||q(x)) = (1-\alpha)KL(p||G_\alpha(p,q)) + \alpha KL(q||G_\alpha(p,q))$$

که در آن $G_\alpha(x,y) = x^{1-\alpha}y^\alpha$ برای $\alpha \in [0,1]$ . به عنوان مثال شکل زیر بیان میکند که این انحراف در تخمین توزیع نسبت به انحراف های اولیه چقدر می تواند بهتر عمل کند. در قسمت (a) توزیع $q(z|x)$ از صفر شدن دوری میکند. در قسمت (b) توزیع $q(z|x)$ هر جایی که p(z) صفر است و در کل واریانس بسیار کمتری نسبت به توزیع اصلی دارد. در شکل (c) حالتی بینابین شکل (a) و (b) رخ داده است. مشکل انحراف .JS این است که این انحراف برای دو توزیع نرمال به صورت تحلیلی محاسبه نمی شود .هم چنین این انحراف یک پارامتر متغیر به نام $\alpha$ دارد که می توان با بازی کردن با مقادیر مختلف آن بین حالت های مختلفی از انحراف جابه جا شد و حالت بهینه را به دست آورد. مقالات دیگری سعی کرده اند تا پارامتر $\alpha$ را به نحو مناسبی بیابند.

### 2.5.4 اضافه کردن یک سری جملات دیگر به ترم Regularization در آموزش VAE ها.

به عنوان مثال یک مقاله به قسمت هموارسازی کران پایین وردشی دو ترم تحت عنوان Diversity و Uncertainty Awareness اضافه کرده بود و نشان داده بود با استفاده از تکنیک drop out در شبکه های عصبی کدکننده و کدگشا می توان این دو ترم جدید اضافه شده در تابع هدف را کنترل کرد و در کل به نتایج بهتری نسبت به حالت های اولیه شد.

به عنوان مثال مقاله ای یکی از چالش های اصلی VAE ها را سوراخ بودن! توزیع پیشین در نظر گرفته بود و ثابت کرده بود که آموزش VAE ها معادل نزدیک کردن توزیع پیشین و توزیع پسین تجمیع شده ( $q(z) = E_{p_d(x)}[q(z|x)]$ ) است. در نتیجه اگر توزیع پیشین نتواند با توزیع پسین تجمیع شده منطبق شود، کیفیت خروجی VAE پایین خواهد بود. در ادامه سعی میکند به آموزش VAE ها بر اساس روش های سنتی، یک گام اضافه کند و مشکل سوراخ بودن توزیع پیشین را حل کند.

مراجع :

[1] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational Inference: A Review for Statisticians. In Journal of the American Statistical Association, 112:518, 859-877, 2017.

[2] C. Zhang, J. Butepage, H. Kjellstrom, and S. Mandt. Advances in variational inference. In IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(8):2008-2026, 2019.

[3] N. Wan, D. Li, and N. Hovakimyan. f-Divergence Variational Inference. In Neural Information Processing Systems (NeurIPS), 2020.

[4] D. Wang, H. Liu, and Q. Liu. Variational inference with tail-adaptive f-divergence. In Advances in Neural Information Processing Systems, 2018.

[5] J. Aneja, A. G. Schwing, J. Kautz, A. Vahdat. A Contrastive Learning Approach for Training Variational Autoencoder Priors. In Advances in Neural Information Processing Systems, 2021.

[6] D. P. Kingma, and M. Welling. Auto-Encoding Variational Bayes. In arriv: 1312.6114, 2014.

[7] J. B. Regli, and R. Silva. Alpha-Beta Divergence For Variational Inference. In arriv: 1805.01045, 2018.

[8] J. Deasy, N. Simidjievski, and P. Liò. Constraining Variational Inference with Geo metric Jensen-Shannon Divergence. In Neural Information Processing Systems, 2020.

[9] D. Wang, H. Liu, and Q. Liu. Variational inference with tail-adaptive f-divergence. In Advances in Neural Information Processing Systems, 2018.