Revisión Bibliográfica

Erika Rivadeneira
erika.rivadeneira@cimat.mx

January 27, 2022

1 Factor Analysis

Factor analysis, análisis factorial en español, es un modelo de variable latente Gaussiano lineal que está fuertemente relacionado probabilisticamente con PCA. Su definición difiere con PCA probabilístico solamente en que la distribución condicional de la variable observable X dada la variable latente z tiene una covarianza diagonal en lugar de isotrópica, de modo que

$$p(X \mid z) = \mathcal{N}(x \mid Wz + \mu, \Sigma),$$

donde Σ es una matriz diagonal $D \times D$. Notemos que el modelo de análisis factorial, en común con PCA probabilístico. asume que las variables observadas $x_1, ..., x_D$, son independientes, dada la variable latente z. Esencialmente, el modelo de análisis factorial explica la estructura de covarianza observada de los datos al representar la varianza independiente asociada con cada coordenada en la matriz Σ y captura la covarianza entre variables en la matriz W. En la literatura sobre análisis factorial, las columnas de W que capturan las correlaciones entre las variables observadas se llaman *factor loadings* y los elementos diagonales de Σ que representan las variaciones de ruido independientes para cada una de las variables, se denominan *uniquenesses* (unicidades).

La distribución marginal de la variable observable X está dada por

$$p(X) = \mathcal{N}\left(X \mid \mu, WW^T + \Sigma\right).$$

Como en PCA probabilístico este modelo es invariante a rotaciones en el espacio latente. Se pueden determinar los parámetros μ, W y Σ del modelo en análisis factorial por máxima verosimilitud. La solución para μ viene nuevamente dada por la media de la muestra. Sin embargo, a diferencia del PCA probabilístico, ya no existe una solución de máxima verosimilitud de forma cerrada para W, que, por lo tanto, debe encontrarse de forma iterativa. Debido a que el análisis factorial es un modelo de variable latente, esto se puede hacer usando un algoritmo **EM** que es análogo al usado para el PCA probabilístico. donde el operador '*diag*' establece todos los elementos no diagonales de una matriz en cero. Otra diferencia entre el PCA probabilístico y el análisis factorial se refiere a su comportamiento diferente bajo las transformaciones del conjunto de datos. Para PCA y PCA probabilístico, si rotamos el sistema de coordenadas en el espacio de datos, obtenemos exactamente el mismo ajuste a los datos pero con la matriz W transformada por la matriz de rotación correspondiente. Sin embargo, para el análisis factorial, la propiedad análoga es que si realizamos un cambio de escala por componentes de los vectores de datos, esto se absorbe en un cambio de escala correspondiente de los elementos de Σ [Bis06].

2 Evidence Lower Bound (ELBO)

Evidence lower bound (límite inferior de la evidencia, ELBO) es una cantidad importante que se encuentra en el núcleo de varios algoritmos de la inferencia probabilística, como la maximización de la esperanza (**EM**) y la inferencia variacional. Para comprender estos algoritmos, es útil comprender ELBO.

Primero, recordemos que en un modelo de variable latente, se postulan a los datos observados X como una realización de alguna variable aleatoria. Además, se postula la existencia de otra variable aleatoria z donde X y z se distribuyen de acuerdo con una distribución conjunta $p(X,z;\theta)$ donde θ parametriza la distribución. Desafortunadamente, los datos son solo una realización de la variable aleatoria, no de z, y por lo tanto z permanece sin ser observado (es decir, latente).

Hay dos tareas predominantes que son de interés:

1. Calcular la distribución posterior $p(z|X,\theta)$, dado un θ fijo.
2. Dado θ desconocido, encontrar la máxima verosimilitud estimada para θ

$$\text{argmax}_\theta \, l(\theta)$$

donde $l(\theta)$ es la función log-verosimilitud:

$$l(\theta) := \log p(X;\theta) = \log \int_z p(X,z;\theta) dz.$$

La inferencia variacional se usa para el propósito 1 y la maximización de esperanzas se usa para el propósito 2. Ambos algoritmos se basan en ELBO.

Para comprender ELBO, primero debemos comprender lo que entendemos por "evidencia". La evidencia, simplemente, es solo un nombre que se le da a la función de verosimilitud evaluada en un θ fijo:

$$\text{evidencia} : = \log p(X;\theta)$$

Intuitivamente, si se elege el modelo correcto p y θ, entonces se espera que la probabilidad marginal de los datos observados X, sea alta. Por lo tanto, un valor más alto de $\log p(X;\theta)$ indica, en cierto sentido, que se puede estar en el camino correcto con el modelo p y los parámetros θ que se han elegido. Es decir, esta cantidad es "evidencia" de que se ha elegido el modelo correcto para los datos.

Además, si se propone que z sigue una distribución denotada por q y que $p(X,z;\theta) := p(X \mid z;\theta)q(z)$, entonces el ELBO es un límite inferior en la evidencia que hace uso de la q conocida (propuesta). Es decir,

$$\begin{aligned}
\log p(X;\theta) &= \log \int p(X,z;\theta) dz \\
&= \log \int p(X,z;\theta) \frac{q(z)}{q(z)} dz \\
&= \log E_{X\sim q}\left[\frac{p(X,z)}{q(z)}\right] \\
&\geq E_{z\sim q}\left[\log \frac{p(X,z)}{q(z)}\right] \quad \text{(Desigualdad de Jensen)}
\end{aligned}$$

donde el ELBO es simplemente el lado derecho de la desigualdad anterior:

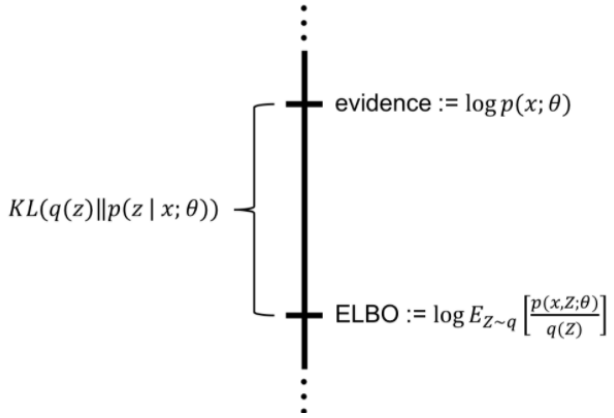
$$ELBO := \mathbb{E}_{z\sim q}\left[\log \frac{p(X,z|\theta)}{q(z)}\right] = \mathcal{L}(q,\theta)$$

2.1 La brecha entre la evidencia y el ELBO

Resulta que la brecha entre la evidencia y el ELBO es precisamente la divergencia de Kullback-Leibler entre $p(z|X,\theta)$ y $q(z)$. Este hecho forma la base del algoritmo de inferencia variacional para la inferencia bayesiana aproximada [Yan07].

La expresión se puede derivar de la siguiente manera:

$$\begin{aligned} KL(q(z)\|p(z \mid X;\theta)) &:= \mathbb{E}_{z\sim q}\left[\log \frac{q(z)}{p(z \mid X;\theta)}\right] \\ &= \mathbb{E}_{z\sim q}[\log q(z)] - \mathbb{E}_{z\sim q}\left[\log \frac{p(X,z|\theta)}{p(X;\theta)}\right] \\ &= \mathbb{E}_{z\sim q}[\log q(z)] - \mathbb{E}_{z\sim q}[\log p(X,z|\theta)] + \mathbb{E}_{z\sim q}[\log p(X|\theta)] \\ &= \log p(X|\theta) - \mathbb{E}_{z\sim q}\left[\log \frac{p(X,z|\theta)}{q(z)}\right] \\ &= \text{ evidence } - \text{ ELBO} \end{aligned}$$



3 Maximización de la esperanza (Algoritmo EM)

El algoritmo EM, en general, es una técnica general para encontrar soluciones de máxima verosimilitud para modelos probabilísticos que tienen variables latentes.

La distribución conjunta $p(X,z|\theta)$ está gobernada por el conjunto de parámetros θ. El objetivo es maximizar la función de verosimilitud dada por

$$p(X \mid \theta) = \sum_z p(X,z \mid \theta).$$

Ahora, sea $q(z)$ una distribución definida sobre todas las variables latentes z, y, se observa que, para cualquier elección de $q(z)$, se cumple la siguiente descomposición:

$$\log p(X \mid \theta) = \mathcal{L}(q,\theta) + KL(q\|p), \tag{1}$$

donde

$$\mathcal{L}(q,\theta) = \sum_z q(z) \ln\left\{\frac{p(X,z \mid \theta)}{q(z)}\right\}$$

$$\mathrm{KL}(q\|p) = -\sum_z q(z) \ln\left\{\frac{p(z \mid X,\theta)}{q(z)}\right\}$$

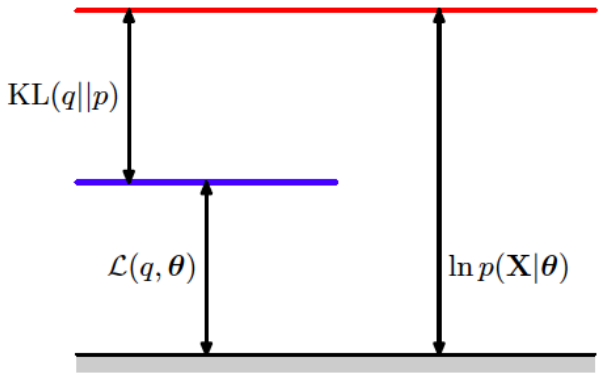


Figure 1: Ilustración de la descomposición dada por (1), que se cumple para cualquier elección de distribución $q(z)$. Debido a que la divergencia de Kullback-Leibler satisface $KL(q||p) \geq 0$, vemos que la cantidad $\mathcal{L}(q,\theta)$ es un límite inferior en la función logarítmica de verosimilitud $\ln p(X|\theta)$.

$\mathcal{L}(q,\theta)$ contiene a la distribución conjunta de X y z mientras que $\mathrm{KL}(q\|p)$ contiene a la distribución condicional de z dado X.

El algoritmo EM es una técnica de optimización iterativa de dos etapas para encontrar soluciones de máxima probabilidad. Suponga que el valor actual de el vector de parámetros es θ^{old} . En el paso **E**, el límite inferior $\mathcal{L}\left(q,\theta^{\text{old}}\right)$ se maximiza con respecto a $q(z)$ mientras se mantiene a θ^{old} fijo. La solución a este problema de maximización se ve facilmente notando que el valor de $\ln p\left(X \mid \theta^{\text{old}}\right)$ no depende de $q(z)$ y así el valor más grande de $\mathcal{L}\left(q,\theta^{\text{old}}\right)$ ocurre cuando la divergencia de Kullback-Leibler desaparece. En otras palabras, $\mathcal{L}\left(q,\theta^{\text{old}}\right)$ se maximiza con respecto a $q(z)$ cuando $q(z)$ es igual a la distribución posterior $p\left(z \mid X,\theta^{\text{old}}\right)$. En este caso, el ELBO es igual a la log-verosimilitud $\ln p(X|\theta)$, ver figura (2)

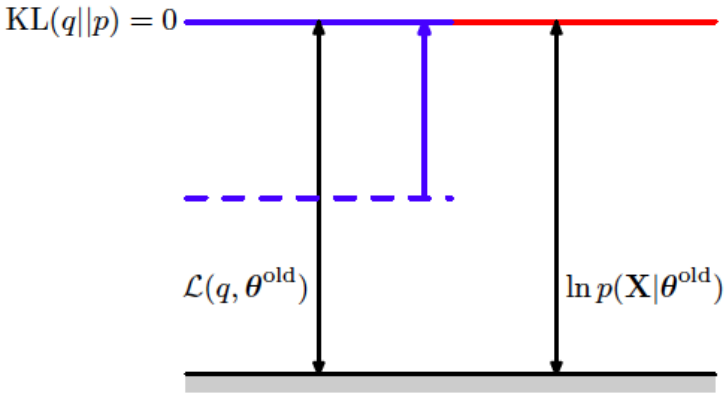


Figure 2: Ilustración del paso **E** de el algoritmo EM. La distribución q es igual a la distribución posterior para los valores actuales de los parámetros θ^{old}, probocando que el ELBO ascienda al mismo valor que la función log-verosimilitud, con la divergencia KL desvaneciéndose.

En el paso **M**, la distribución $q(z)$ está fija y el ELBO $\mathcal{L}(q,\theta)$ se maximiza con respecto a θ dando así un valor θ^{new} . Esto probocará que el ELBO $\mathcal{L}$ incremente (al menos que ya esté en su máximo), lo cual hará que la función log-verosimilitud incremente. Debido a que la distribución q se determina utilizando los valores de los parámetros antiguos en lugar de los nuevos valores y se mantiene fija durante el paso de M, no será igual a la nueva distribución posterior $p\left(z \mid X,\theta^{\text{new}}\right)$, y por lo tanto habrá una divergencia de KL distinta de cero. Por lo tanto, el incremento en la función log-verosimilitud es mayor al incremento en el ELBO, ver figura (3).

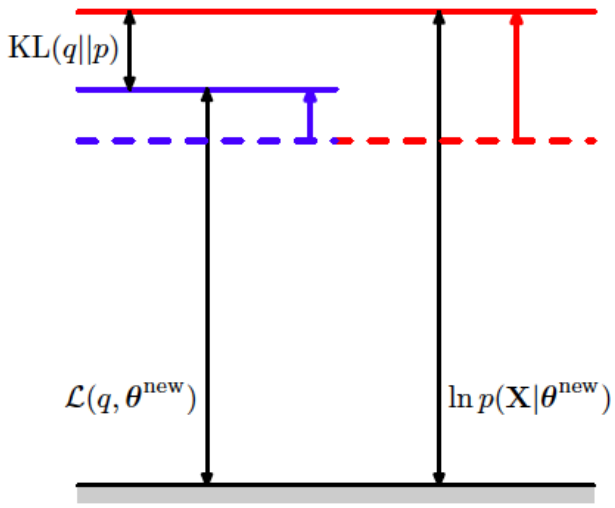


Figure 3: Ilustración del paso **M** del algoritmo EM. La distribución $q(z)$ está fija y el ELBO $\mathcal{L}(q,\theta)$ se maximiza con respecto al vector θ dando un valor θ^{new}. Dado que la divergencia KL es no negativa la función log-verosimilitud $\ln p(X|\theta)$ incremente por lo menos to al menos tanto como el ELBO lo hace.

3.1 EM para Factor Analysis

Las ecuaciones del paso **E** están dadas por

$$\mathbb{E}[\mathbf{z}_n] = GW^T\Psi^{-1}(x_n - \bar{x}),$$
$$\mathbb{E}\left[z_n z_n^T\right] = G + \mathbb{E}[z_n]\mathbb{E}[z_n]^T,$$

donde

$$G = \left(I + W^T\Psi^{-1}W\right)^{-1}.$$

Esto se deriva a través de la regla de Bayes para Gaussianas. De manera similar, las ecuaciones del paso **M** toman la forma

$$\mathbf{W}^{\text{new}} = \left[\sum_{n=1}^{N}(x_n - \bar{x})\,\mathbb{E}[z_n]^T\right]\left[\sum_{n=1}^{N}\mathbb{E}\left[z_n z_n^T\right]\right]^{-1},$$
$$\mathbf{\Psi}^{\text{new}} = \text{diag}\left\{S - W_{\text{new}}\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}[z_n](x_n - \bar{x})^T\right\}$$

4 Inferencia Variacional

Primero, se discutirá cómo la optimización variacional puede ser aplicada al problema de inferencia. Supongamos que se tiene un modelo Bayesiano completo donde todos los parámetros están dados por distribuciones a priori dadas. El modelo puede contener variables latentes, se denota por z al conjunto de todas las variables latentes y parámetros. El conjunto de variables observadas se denota por X. Nuestro modelo probabilístico satisface la distribución conjunta $p(X,z)$ y el objetivo es encontrar una aproximación de la distribución posterior $p(z|X)$ así como la evidencia del modelo $p(X)$. Como en la discusión de EM, se puede descomponer el logaritmo de la distribución marginal usando

$$\log p(X) = \mathcal{L}(q,\theta) + \text{KL}(q\|p),$$

donde

$$\mathcal{L}(q,\theta) = \int q(z) \log \left\{ \frac{p(X,z)}{q(z)} \right\} dz, \tag{2}$$

$$\mathrm{KL}(q\|p) = -\int q(z) \ln \left\{ \frac{p(z \mid X)}{q(z)} \right\} dz. \tag{3}$$

Esto difiere de la discusión de EM solamente en que el vector de parámetros θ ya no aparece, porque ahora los parámetros son variables estocásticas y son absorbidas en z. Como antes, se puede maximizar el ELBO $\mathcal{L}(q,\theta)$ optimizando con respecto a la distribución $q(z)$, lo cual es equivalente a minimizar la divergencia KL. Si se permite cualquier elección de $q(z)$, entonces el máximo del ELBO ocurre cuando la divergencia KL desaparece, lo cual sucede cuando $q(z)$ es igual a la distribución posterior $p(z|X)$. Sin embargo, se supondrá que el modelo es tal que trabajar con la distribución posterior verdadera es intratable.

Por lo tanto, se considera en cambio una familia restringida de distribuciones $q(z)$ y luego se busca el miembro de esta familia para el cual se minimiza la divergencia KL. El objetivo es restringir la familia lo suficiente para que comprendan solo distribuciones manejables, y al mismo tiempo permitir que la familia sea lo suficientemente rica y flexible como para proporcionar una buena aproximación a la verdadera distribución posterior. En particular, no hay "sobreajuste" asociado con distribuciones altamente flexibles. El uso de aproximaciones más flexibles simplemente nos permite acercarnos más a la verdadera distribución posterior.

Una manera de restringir la familia de distribuciones aproximadas es al usar una distribución paramétrica $q(z \mid \omega)$ governada por el conjunto de parámetros ω. El ELBO $\mathcal{L}(q,\theta)$ entonces estaría en función de ω, y se puede aprovechar las técnicas de optimización no lineal estándar para determinar los valores óptimos de los parámetros.

Para ver que la inferencia variacional funciona se argumenta de manera similar a EM. Cuando las esperanzas se calculan utilizando los valores actuales para las variables que no se actualizan, se establece implícitamente la divergencia KL entre las distribuciones de ponderación y las distribuciones posteriores en 0. Luego, la actualización aumenta la función log-verosimilitud.

4.1 De EM a Inferencia Variacional

- En EM se maximiza alternadamente el ELBO con respecto a θ y la distribución de probabilidad q
- En la inferencia variacional, se descarta la distinción entre variables latentes y parámetros de una distribución, es decir. se reemplaza $p(X,z|\theta)$ con $p(X,z)$. Efectivamente, esto pone una distribución de probabilidad en los parámetros θ, luego los absorbe en z.
- Tratamiento totalmente Bayesiano en lugar de una estimación puntual de los parámetros.

4.2 Distribuciones Factorizadas

Se considera una alternativa para restringir la familia de distribuciones $q(z)$. Suponga que se particiona los elementos de z en grupos disjuntos denotados por z_i donde $i = 1,\ldots,M$. Entonces, asumiendo que las distribuciones q se factorizan con respecto a estos grupos, así

$$q(z) = \prod_{i=1}^{M} q_i(z_i). \tag{4}$$

Esta forma factorizada de inferencia variacional corresponde a un marco de aproximación desarrollado en física llamado *mean file theory* [Par98]. Entre todas las distribuciones $q(z)$ que tienen la forma (4), ahora se busca la distribución para el cual ELBO $\mathcal{L}(q,\theta)$ es mayor. Por lo tanto, se busca hacer una optimización variacional de $\mathcal{L}(q,\theta)$ con respecto a todas las distribuciones $q_i(z_i)$, lo cual se logra optimizando con respecto a cada uno de los factores a la

vez [Bis06]. Para lograr este cometido se reemplaza (4) en (11) obteniendo

$$\begin{aligned}
\mathcal{L}(q,\theta) &= \int q(z)\log\left[\frac{p(X,z)}{q(z)}\right]dz = \int q(z)\log p(X,z) - q(z)\log q(X)dz \\
&= \int \prod_i q_i(z_i)\left\{\log p(X,z) - \sum_k \log q_k(z_k)\right\}dz \\
&= \int q_j(z_j)\left\{\int \prod_{i\neq j} q_i(z_i)\left\{\log p(X,z) - \sum_k \log q_k(z_k)\right\}dz_i\right\}dz_j \\
&= \int q_j(z_j)\left\{\int \log p(X,z)z_i dz_i - \int \prod_{i\neq j}\sum_k q_i(z_i)\log q_k(z_k)\,dz_i\right\}dz_j \\
&= \int q_j(z_j)\left\{\int \log p(X,z)\prod_{i\neq j} q_i(z_i)\,dz_i - \log q_j(z_j)\int \prod_{i\neq j} q_i(z_i)\,dz_i\right\}dz_j + \text{ const} \\
&= \int q_j(z_j)\left\{\log p(X,z)\prod_{i\neq j} q_i(z_i)\,dz_i\right\}dz_j - \int q_j(z_j)\log q_j(z_j)\,dz_j + \text{ const} \\
&= \int q_j(z_j)\,\mathbb{E}_{i\neq j}[\log p(X,z)]dz_j - \int q_j(z_j)\log q_j(z_j)\,dz_j + \text{const}.
\end{aligned}$$

Entonces, la actualización de mean file es

$$q_j(z_j)^{(t)} \quad \leftarrow \text{argmax}_{q_j(z_j)} \int q_j(z_j)\,\mathbb{E}_{i\neq j}[\log p(X,z)]dz_j - \int q_j(z_j)\log q_j(z_j)\,dz_j.$$

La idea en general no son las ecuaciones de actualización en sí mismas, sino en congelar algunas de las variables, calcular las esperanzas sobre estas variables y actualizar las variables sobrantes utilizando estas esperanzas.

5 Modelos de Variables Latentes

Antes de mencionar en qué se basan los modelos de variables latentes primero se introduce la idea central de los modelos generativos. Los modelos generativos permiten a una máquina "aprender" los patrones que existen en los datos con los que son entrenadas y a partir de dicho aprendizaje, son capaces de generar datos similares que en algunos casos pueden ser casi tan "reales" como los que se utilizaron inicialmente para su entrenamiento. Al entrenar un modelo generativo, cuanto más complicadas son las dependencias entre las dimensiones, más difíciles son los modelos de entrenar [Doe21].

Ahora, digamos que se tiene un vector de variables latentes z en un espacio Z de alta dimensión que podemos muestrear fácilmente de acuerdo con alguna función de densidad de probabilidad (PDF) $p(z)$ definida sobre Z, es decir, para cada punto de datos X en el conjunto de datos, hay una (o varias) configuraciones de las variables latentes z que hacen que el modelo genere algo muy similar a X. Entonces, digamos que tenemos una familia de funciones $f(z;\theta)$, parametrizada por un vector θ en algún espacio Θ, donde $f : Z \times \Theta \to X$ [Doe21].

f es determinista, pero si z es aleatoria y θ está fija, entonces $f(z,\theta)$ es una variable aleatoria en el espacio X. Se busca optimizar θ de tal manera que se pueda muestrear z de $p(z)$, con alta probabilidad. Es decir, el objetivo es es maximizar la probabilidad de cada X en el conjunto de entrenamiento bajo todo el proceso generativo, de acuerdo con:

$$P(X) = \int P(X \mid z;\theta)P(z)dz. \tag{5}$$

En (5), $f(z;\theta)$ es reemplazado por la distribución $P(X \mid z;\theta)$, lo cual permite hacer explícita la dependencia de X en z mediante el uso de la ley de probabilidad total. Este marco es conocido como "maximum likelihood" (en español, probabilidad máxima). En los variational autoencores (VAEs) la elección de esta distribución, $p(X|z;\theta)$, suele ser una Gaussiana, i.e., $P(X \mid z;\theta) = \mathcal{N}\left(X \mid f(z;\theta), \sigma^2 * I\right)$. Esto es, que su media es $f(z;\theta)$ y covarianza igual a la matriz

identidad I multiplicada por un escalar σ (hiperparámetro). Este reemplazo es necesario para formalizar la intuición de que algún z necesita dar como resultado muestras que son similares a X. Al tener una distribución Gaussiana, se puede usar el descenso por gradiente (o cualquier otra técnica de optimización) para incrementar $p(X)$ haciendo que $f(z;\theta)$ se acerque a X para algún z [Doe21].

6 Variational Autoencoders

Se pueden considerar a los VAEs como una versión probabilística de un autocodificador determinista. Los VAEs aproximadamente maximizan la ecuación (5). Se denominan "autoencoders" solo porque el objetivo del entrenamiento final que se deriva de esta configuración tiene un encoder y un decoder, y se asemeja a un autoencoder tradicional. La principal ventaja es que un VAE es un modelo generativo que puede crear nuevas muestras, mientras que un autoencoder solo calcula incrustaciones de vectores de entrada.

Para resolver la ecuación (5), hay dos problemas con los que los VAEs deben lidear: cómo definir las variables latentes z (i.e., decidir qué información representan), y cómo lidear con la integral sobre z. VAEs provee una respuesta definitiva a ambas incógnitas.

Para solucionar el primer problema los VAEs suponen que no hay una interpretación simple de las dimensiones de z, y en su lugar afirman que las muestras de z se pueden extraer de una distribución simple, es decir, $\mathcal{N}(0,I)$, donde I es la matriz de identidad. Para justificar esto, la clave es notar que cualquier distribución en d dimensiones se puede generar tomando un conjunto de d variables que están normalmente distribuidas y mapeándolas a través de una función suficientemente complicada [Dev86].

Ahora, para resolver la segunda incógnita, maximizar la ecuación (5), donde $p(z) = \mathcal{N}(z|0,I)$ varios modelos de aprendizaje de máquina utilizan una métrica de similitud, pero en la práctica son difíciles de diseñar en dominios complejos como la visión, y son difíciles de entrenar sin etiquetas que indiquen qué puntos de datos son similares entre sí. En cambio, los VAEs alteran el procedimiento de muestreo para hacerlo más rápido, sin cambiar la métrica de similitud.

6.1 Configuración de la función objetivo

La idea clave detrás los VAEs es intentar muestrear valores de z que probablemente hayan producido X, y calcular $p_\theta(X)$ a partir de ellos. Esto significa que necesitamos una nueva función $q(z|X)$ que puede tomar un valor de X y dar una distribución sobre los valores de z que probablemente produzcan X. Se espera que el espacio de los valores de z que probablemente estén bajo q sea mucho más pequeño que el espacio de todas las z que probablemente estén bajo la priori $p(z)$. Esto permite, por ejemplo, calcular $\mathbb{E}_{z\sim q}[p(X|z)]$ relativamente fácil.

Para entender la relación entre $\mathbb{E}_{z\sim q}[p(X|z)]$ y $p(X)$ se comienza recordando la definición de la divergencia Kullback-Leibler entre $p(z|X)$ y $q(z)$, para una q arbitratia:

$$KL(q(z)||p(z|X)) = \mathbb{E}_{z\sim q}[\log q(z) - \log p(z|X))].$$

Se pueden obtener $p(X)$ y $p(X|z)$ en esta ecuación aplicando la regla de Bayer a $p(z|X)$. Así, se obtiene:

$$KL(q(z)||p(z|X)) = \mathbb{E}_{z\sim q}[\log q(z) - \log p(X|z)) - \log p(z)] + \log p(X).$$

Aquí, $\log p(X)$ sale fuera de la esperanza dado que no depende de z. Luego, multiplicando por -1 ambos lados de la ecuación, reordenando términos y contrayendo la parte de $\mathbb{E}_{z\sim q}$ en términos de la divergencia KL se obtiene:

$$\log p(X) - KL(q(z)||p(z|X)) = \mathbb{E}_{z\sim q}[\log p(X|z)] - KL(q(z)||p(z)).$$

Notemos que X está fija, y q puede tener cualquier distribución. Como es de interés inferir $p(X)$, tiene sentido construir q de tal manera que dependa de X, y en particular, alguna construcción de q que haga que $KL(q(z)||p(z|X))$ sea pequeña:

$$\log p_\theta(X) - KL(q(z|X)||p(z|X)) = \mathbb{E}_{z\sim q}[\log p_\theta(X|z)] - KL(q(z|X)||p(z)). \tag{6}$$

La ecuación (6) sirve como el núcleo del VAE. El lado izquierdo de la ecuación tiene la cantidad que queremos maximizar: $logp(X)$ (más un término de error, lo que hace que q produzca z's las cuales reproducen una X dada) y simultáneamente se minimiza $KL(q(z|X)||p(z|X))$. $p(z|X))$ no se lo puede calcular analíticamente: describe los valores de z que probablemente den lugar a una muestra similar a X en nuestro modelo. Sin embargo, el segundo término, de este lado de la ecuación, empuja a $q(z|X)$ para que se parezca a $p(z;X)$. El lado derecho se puede optimizar a través del descenso de gradiente estocástico dada la elección correcta de q [Doe21].

6.2 Optimizando la función objetivo

La elección usual es decir que $q(z|X) = \mathcal{N}(z|\mu(X;\theta),\Sigma(X;\theta))$, donde μ y Σ son funciones deterministas con parámetros θ que pueden aprender de los datos. En la práctica, μ y Σ se implementan a través de redes neuronales, y Σ está restringido a ser una matriz diagonal. El último término de la ecuación (6), $KL(q(z|X)||p(z))$, es ahora una divergencia KL entre dos distribuciones Gaussianas multivariadas, la cual puede ser calculada en forma cerrada como:

$$KL(\mathcal{N}(\mu_0,\Sigma_0)||\mathcal{N}(\mu_1,\Sigma_1)) = \frac{1}{2}\left(\mathrm{tr}\left(\Sigma_1^{-1}\Sigma_0\right) + (\mu_1-\mu_0)^\top \Sigma_1^{-1}(\mu_1-\mu_0) - k + \log\left(\frac{\det\Sigma_1}{\det\Sigma_0}\right)\right), \tag{7}$$

donde k es la dimensionalidad de la distribución. En nuestro caso, la ecuación (7) se simplifica de la siguiente manera:

$$KL(\mathcal{N}(\mu(X;\theta),\Sigma(X;\theta))||\mathcal{N}(0,I)) = \frac{1}{2}\left(\mathrm{tr}(\Sigma(X;\theta)) + (\mu(X;\theta))^\top(\mu(X;\theta)) - k - \log\det(\Sigma(X;\theta))\right).$$

Por otro lado, para estimar el primer término del lado derecho de la ecuación (6), $\mathbb{E}_{z\sim q}[\log p_\theta(X|z)]$, se puede hacer uso de algún método de muestreo, pero obtener una buena estimación requeriría pasar muchas muestras de z a f, lo que sería computacionalmente costoso. Por lo tanto, como es estándar en el descenso de gradiente estocástico, se toma una muestra de z y se trata $\log p_\theta(X|z)$ para z como una aproximación de $\mathbb{E}_{z\sim q}[\log p_\theta(X|z)]$. La ecuación completa que se desea optimizar es

$$\mathbb{E}_{X\sim D}\left[\log p_\theta(X) - KL(q(z|X)||p(z|X))\right] = \mathbb{E}_{X\sim D}\left[\mathbb{E}_{z\sim q}[\log p_\theta(X|z)] - KL(q(z|X)||p(z))\right],$$

donde, D representa al conjunto de datos. Si se considera el gradiente de esta ecuación, por la linealidad de los operadores el gradiente se puede introducir en la esperanza. Por lo tanto, se puede muestrear un solo valor de X y un solo valor de z de la distribución $q(z|X)$, y calcular el gradiente de

$$\log p_\theta(X|z) - KL(q(z|X)||p(z)). \tag{8}$$

La red descrita en la ecuación (8) se parece al de la gráfica del lado derecho de la figura (4). El pase directo de esta red funciona bien y, si la salida se promedia sobre muchas muestras de X y z, produce el valor esperado correcto. Sin embargo, se necesita retropropagar (en inglés back-propagate) el error a través de una capa que muestrea z de $q(z|X)$, la cual es una operación no continua y no tiene gradiente.

El descenso de gradiente estocástico mediante retropropagación puede manejar entradas estocásticas, pero no unidades estocásticas dentro de la red. La solución en [KW14], llamada ***"truco de reparametrización"*** , es mover el muestreo a una capa de entrada.

Dado $\mu(X;\theta)$ y $\Sigma(X;\theta)$, la media y covarianza de $q(z|X)$, se puede muestrear de $\mathcal{N}(\mu(X;\theta),\Sigma(X;\theta))$ muestreando primero $\varepsilon \sim \mathcal{N}(0,I)$, y luego calcular

$$z = \mu(X;\theta) + \Sigma^{1/2}(X;\theta) * \varepsilon.$$

Así, la ecuación de la que realmente tomamos el gradiente es:

$$\mathbb{E}_{X\sim D}\left[\mathbb{E}_{\varepsilon\sim\mathcal{N}(0,I)}\left[\log p_\theta(X|z = \mu(X;\theta) + \Sigma^{1/2}(X;\theta) * \varepsilon\right)] - KL(q(z|X)||p(z))\right]. \tag{9}$$

Este truco de reparametrización se muestra esquemáticamente en el gráfico derecho de la figura (4). Notemos que ninguna de las esperanzas de la ecuación (9) se encuentran con respecto a las distribuciones que dependen de los

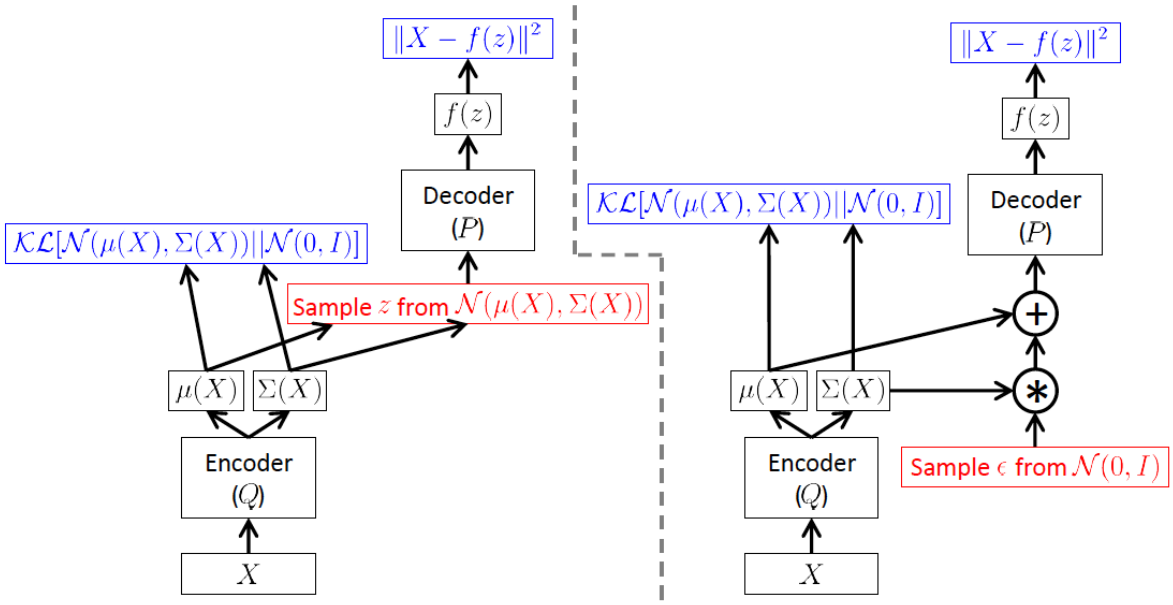


Figure 4: Gráfico para cómputo de un VAE, donde $p(X|z)$ es Gaussiano. Los recuadros rojos muestran las operaciones de muestreo que no son diferenciables. Los recuadros azules muestran capas de pérdida (se asumen probabilidades y a priori Gaussianas). a) Sin truco de reparametrización. b) Con truco de reparametrización. Los gradientes pueden fluir desde la pérdida de salida, de regreso a través del decoder y hacia el encoder.

parámetros de nuestro modelo, por lo que por linealidad de los operadores se puede introducir el gradiente dentro de las esperanzas manteniendo la igualdad.

Al momento de probar el modelo, cuando se desea generar nuevas muestras, se introducen valores de $z \sim \mathcal{N}(0, I)$ en el decoder. Es decir, se elimina el encoder, incluidas las operaciones de multiplicación y suma que cambian la distribión de z. Esta red se muestra en la figura [5].

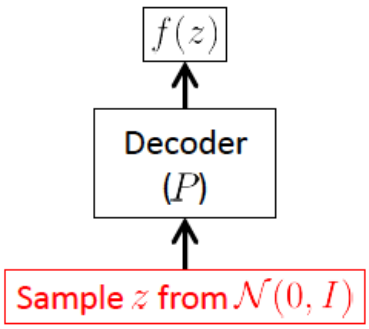


Figure 5: El The testing-time de un autoencoder variacional, el cual permite generar nuevas muestras. La red del encoder se descarta.

Notemos que el lado derecho de la ecuación (6) es un límite inferior de $p_\theta(X)$, i.e., es el ELBO, denotado por

$$\mathcal{L}(q, \theta) = \mathbb{E}_{z \sim q}[\log p_\theta(X|z)] - KL(q(z|X)||p(z)).$$

Esto debido a que $KL(q(z|X)||p(z|X))$ es siempre positivo, esto implica que

$$\mathcal{L}(q,\theta) \leq \mathbb{E}_{z\sim q}[\log p_\theta(X|z)],$$

es decir, lo que significa que $\mathcal{L}(\theta|X)$ es un límite inferior de la log-probabilidad esperada de las observaciones. En otras palabras, el límite inferior $\mathcal{L}(\theta|X)$ alcanza la probabilidad logarítmica si la distribución aproximada está perfectamente cerrada a la distribución posterior verdadera [Yan07].

Finalmente, hay que observar que el ELBO todavía no se puede calcular en forma cerrada debido a la esperanza sobre z, la cual requiere muestreo. Sin embargo, el muestreo de z de q proporciona un estimador para la esperanza que generalmente converge mucho más rápido que el muestreo de z de $\mathcal{N}(0,I)$. Por lo tanto, este límite inferior puede ser una herramienta útil para tener una idea aproximada de qué tan bien el modelo está capturando un punto de datos particular X.

6.3 Comparación entre VAEs y autoencoders

La principal ventaja de un VAE es que se puede utilizarlo para generar nuevos datos de ruido aleatorio. En particular, se muestrea z de la a priori Gaussiana $\mathcal{N}(z|0,I)$, y luego pasa por el decoder para obtener $\mathbb{E}[p_\theta(X|z)]$. El decoder de un VAE se entrena para convertir puntos aleatorios en el espacio de incrustación (embedding space), generando perturbaciones en los inputs codificados, a outputs sensibles. En contraste, el decoder de un autoencoder determinista solo obtiene como entradas las codificaciones exactas del conjunto de entrenamiento, por lo que no sabe qué hacer con entradas aleatorias que están fuera de lo que fue entrenado. Por tanto, un autoencoder estándar no puede crear nuevas muestras.

7 Aprendizaje Local

La familia de enfoques señala directamente las cantidades más críticas para las clasificaciones, mientras que cualquier otra información menos irrelevante para este propósito simplemente se omite. En comparación con el aprendizaje global, no se asume ningún modelo y tampoco se incluirá información global explícita en este esquema. Entre estos de métodos se encuentran las redes neuronales ([Fau94], [Hay99]), los métodos Gabriel Graph [BDH96], los clasificadores de gran margen [CS00], incluida la máquina de vectores de soporte (SVM).

Los modelos neuronales de recomendación ([Beu+17], [Che+16], [CAS16], [He+18]) han sido estudiados para representar factores complexos y no lineales entre las interacciones usuario-item. Aunque las redes neuronales profundas (DNNs) son capaces de identificar los patrones complejos de las interacciones usuario-item, en estudios recientes ([DCJ19], [Lud+19])se reportan que la ganancia de rendimiento en el problema de recomendación es menos innovadora que en otros dominios como la visión artificial, reconocimiento de voz o procesamiento del lenguaje natural. Pueden existir varias razones por las que esto suceda, tales como escasez extrema de datos (sparcity), ambigüedad de datos por comentarios faltantes y comentarios ruidosos de los usuarios.

Choi et al. (2021) proponen la hipótesis de que como las DNN adoptan una arquitectura de red profunda y capas de activación no lineales y, por lo tanto, generalmente son más potentes que los modelos tradicionales; sin embargo, los modelos de recomendación neuronal suelen emplear redes poco profundas debido a la escasez de datos de entrenamiento. Una arquitectura tan superficial con esquemas de entrenamiento tradicionales solo está limitada en la identificación de diversos patrones locales de interacciones usuario-item y en su mayoría está sesgada por el patrón global.

Lee et al. (2013) proponen una suposición local de bajo rango (Figura 6) relajando significativamente la contraparte global, según la cual: "Una matriz de calificación no se descompone necesariamente en matrices globales de bajo rango, y es una unión de múltiples matrices locales de bajo rango compuestas por un subconjunto de usuarios-items que comparten intereses locales". Por ejemplo, en el dominio de películas, un subgrupo de usuarios que prefieren las películas de comedia romántica es de rango bajo; otro subgrupo de usuarios que prefieren películas de ciencia ficción en la década de 2010 también se encuentra en un rango bajo; sin embargo, una unión de estos puede no estar en rango bajo. Por lo tanto, se puede suponer que los factores dominantes que determinan la preferencia de cada grupo son bastante diferentes en estos dos subgrupos, por ejemplo, la calidad de los gráficos por computadora, que puede importar solo en la categoría de ciencia ficción.

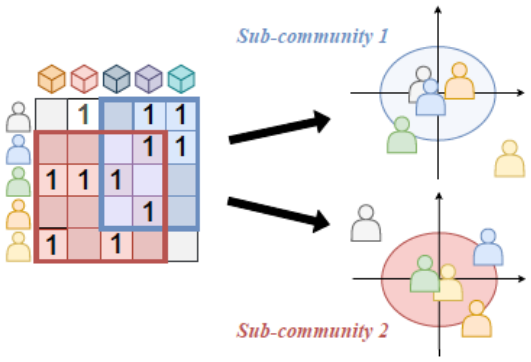


Figure 6: Ilustración de la suposición local de bajo rango. Dos submatrices son localmente de bajo rango, en ellas se pueden descubrir diferentes subcomunidades.

Por otro lado, los modelos locales existentes no exploran completamente el potencial del supuesto local de rango bajo dado que: primero, Christakopoulou and Karypis, 2016, Christakopoulou and Karypis, 2018, Lee et al., 2013 y Lee et al., 2016 no descubren completamente la localidad diversa. Lee et al., 2013 y Lee et al., 2016 adoptan una cobertura relativamente amplia de usuarios en el modelo local, el tamaño de los modelos locales es cercano al del modelo global, violando así la intuición de los modelos locales. Aunque Christakopoulou and Karypis emplearon los modelos locales con pequeñas subcomunidades, su desempeño fue a menudo peor que el del modelo global debido a la falta de datos de entrenamiento. Segundo, los modelos locales existentes no intentan entrenar y combinar una gran cantidad de modelos locales para capturar subcomunidades más pequeñas y coherentes. Por lo tanto, se puede considerar que los estudios previos desarrollaron un pequeño número de modelos relativamente similares. Finalmente, los modelos básicos para aprender los modelos locales a menudo se limitaban a modelos de factores latentes lineales, lo que dificultaba la identificación de patrones no lineales significativos en los modelos locales.

Para solventar estos inconvenientes, Choi et al. (2021) proponen un modelo local, llamado Local Collaborative Au-

toencoders (LOCA). Este modelo proporciona una arquitectura generalizada para aprender una variedad de modelos locales al identificar varias subcomunidades para entrenamiento e inferencia. Además, LOCA puede manejar una gran cantidad de subcomunidades pequeñas y coherentes para el modelo local y el modelo está basado en un autoencoder como su modelo base. Los modelos basados en autoencoders tienen capas de activación no lineales para representar los patrones no lineales significativos en el modelo local.

7.1 Modelos locales con factores latentes

Dados un conjunto $\mathcal{U}$ de m usuarios y un conjunto $\mathcal{I}$ de n items, se tiene una matriz binaria $\mathrm{R} \in \{0,1\}^{m \times n}$. Una entrada $r_{ui} \in \mathrm{R}$ representa una calificación implícita por usuario $u \in \mathcal{U}$ en el item $i \in \mathcal{I}$. Si $r_{ui} = 1$, significa que tiene una calificación positiva; de otro modo, se muestra una calificación faltante. Dado un usuario $u, I_u^+ = \{i \in \mathcal{I} \mid r_{ui} = 1\}$ y $I_u^- = I - I_u^+$ son un conjunto de items con calificaciones positivas y faltantes, respectivamente.

Luego, dada una matriz de calificaciones R, primero se entrena un modelo de recomendación $M(\mathrm{R};\theta) : \{0,1\}^{m \times n} \rightarrow \mathbb{R}^{m \times n}$ con parámetro θ para inferir una calificación $\hat{r}_{ui}$ para el usuario u en el item $i \in I_u^-$. Un modelo local $M^{\text{local}}\left(\mathrm{R};\theta^{(j)}\right)$ es entrenado con su peso correspondiente. Formalmente, cada modelo local tiene un peso correspondiente $\mathcal{T} = \left\{\mathrm{T}^{(1)},\ldots,\mathrm{T}^{(q)}\right\}$, donde $\mathrm{T}^{(j)} \in \mathbb{R}^{m \times n}$ (para $j = 1,\ldots,q$) representa la importancia de cada par (usuario, item) en la matriz de calificaciones R. Después de entrenar todos los modelos locales, R es aproximada por la suma de los múltiples modelos locales. Así,

$$\hat{\mathrm{R}} = \sum_{j=1}^{q} \mathrm{T}^{(j)} \odot M^{\text{local}}\left(\mathrm{R};\theta^{(j)}\right) \oslash \mathrm{T},$$

donde, $\mathrm{T} = \sum_{j=1}^{q} \mathrm{T}^{(j)}$, y $\odot$ & $\oslash$ son el producto y la división por elementos, respectivamente.

Christakopoulou et al. (2018) proponen un método que integra un modelo global con múltiples modelos locales. Sea $M^{\text{global}}\left(\mathrm{R};\theta^{(g)}\right)$ la notación del modelo global. Entonces,

$$\hat{\mathrm{R}} = \alpha M^{\text{global}}\left(\mathrm{R};\theta^{(g)}\right) + (1-\alpha)\sum_{j=1}^{q} \mathrm{T}^{(j)} \odot M^{\text{local}}\left(\mathrm{R};\theta^{(j)}\right) \oslash \mathrm{T},$$

donde α es un hyper-parámetro de control de importancia del modelo global.

7.2 Local Collaborative Autoencoders (LOCA)

Choi et al. (2021) exponen la hipótesis de que el tamaño óptimo del vecindario puede ser diferente en las etapas de entrenamiento e inferencia. Esto se debe a que el entrenamiento, por naturaleza, tiende a beneficiarse de datos más extensos. De acuerdo con los enfoques convencionales del vecino más cercano ([Gol+92], [Her+99], [Sar+01]), se utilizan más vecinos para el entrenamiento y se enfoca en un vecindario más pequeño con usuarios objetivo fuertemente conectados para la inferencia. Esta estrategia puede considerarse como un aumento de datos para mejorar la formación de modelos locales, que representan subcomunidades pequeñas y coherentes. Sin embargo, a diferencia del aumento de datos existente que genera usuarios virtuales de forma sintética, se eligen usuarios reales del vecindario en toda la matriz.

El modelo de Choi et al. se basa en un enfoque de divide-and-conquer con tres pasos:

1. Descubrir un conjunto de comunidades locales (dividir).
2. Entrenar un modelo local para cada subcomunidad (conquistar).
3. Inferir usuario preferencias combinando el modelo global y múltiples modelos locales (agregación).

La Figura (7) ilustra el marco de LOCA.

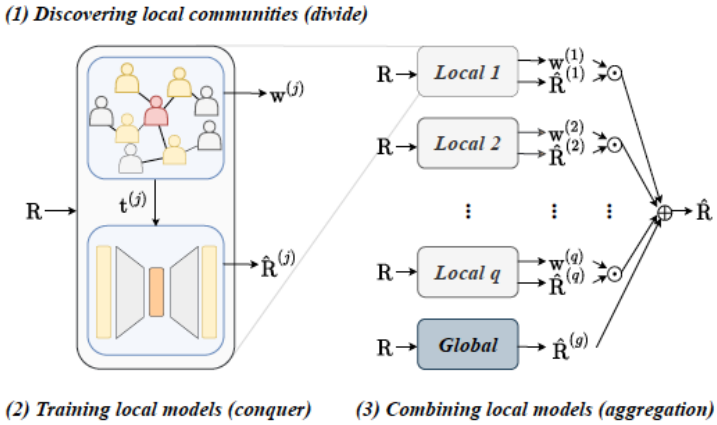


Figure 7: Arquitectura del modelo LOCA. $\mathcal{T}$ y $\mathcal{W}$ se los utilizan para entrenar e inferir los modelos locales. Ambos pesos se calculan en base a las similaridades de cada usuario con el usuario ancla.

8 Variational Autoencoders for Collaborative Filtering

En sistemas de recomendación se observa cómo es la interacción entre usuarios y un conjunto de ítems. Con estos datos, buscamos mostrar a los usuarios un conjunto de ítems, que no hayan visto, que les gustarán. Una técnica de los sistemas de recomendación para lograr este cometido es el filtrado colaborativo (collaborative filtering). El filtrado colaborativo predice qué ítems preferirá un usuario al descubrir y explotar los patrones de similitud entre usuarios y los ítems.

En la actualidad se está aplicando redes neuronales al filtrado colaborativo con resultados prometedores. Esto debido al agregar características no lineales, cuidadosamente diseñadas, en los modelos de factor latente lineal puede aumentar significativamente el rendimiento de las recomendaciones. Una red neuronal que permite explorar modelos probabilísticos de variables latentes no lineales, en un conjunto de datos de recomendación a gran escala, son los variational autoencoders (VAEs). Los VAEs generalizan modelos lineales de factores latentes.

Dawen Liang y Rahul G. Krishnan muestran que las probabilidades multinomiales son adecuadas para modelar datos de retroalimentación implícita y son un proxy más cercano a la pérdida de clasificación en relación con funciones de probabilidad más populares, como gaussiana y logística. Además, dado que el conjunto de datos en sistemas de recomendación es 'pequeño', dado que los usuarios interactuan con una pequeña proporción de ítems, para aprovechar las calificaciones escasas de los usuarios y evitar el sobreajuste, se construye un modelo probabilístico de variables latentes que comparte la fuerza estadística entre los usuarios y los elementos. Los autores muestran que emplear un enfoque bayesiano basado en ciertos principios es más robusto independientemente de la escasez de datos.

Para evaluar los sitemas de recomendación se suelen utilizar medidas basadas en rankings como la métrica *Recall* y la métrica discounted cumulative gain (DCG). Ambas métricas comparan la clasificación predecida de los ítems con su clasificación real. La métrica *Recall@R* es la fracción de ítems relevantes que son recomendados, esta métrica considera que todos los elementos, de los primeros R, clasificados son igualmente importantes. Formalmente, se define a $\omega(r)$ como la clasificación de r, $\mathbb{I}[\cdot]$ es la función indicadora, y I_u es el conjunto de ítems que el usuario u calificó. Entonces, *Recall@R* para el usuario u es

$$\text{Recall@R } (u,\omega) := \frac{\sum_{r=1}^{R} \mathbb{I}[\omega(r) \in I_u]}{\min(M, |I_u|)}.$$

Mientras que DCG es una métrica que penaliza si un ítem relevante aparece bajo en el ranking. Formalmente, DCG@R está definida por

$$\text{DCG@}R(u,\omega) := \sum_{r=1}^{R} \frac{2^{\mathbb{I}[\omega(r)\in I_u]} - 1}{\log(r+1)}.$$

NDCG@R es el DCG@R linealmente normalizada de $[0, 1]$.

9 Metodología

9.1 Modelo

Los Deep latent Gaussian models (DLGMs) [RMW14] son una clase general de modelos gráficos dirigidos profundos que consisten en variables latentes gaussianas en cada capa de una jerarquía de procesamiento. El modelo consta de L capas de variables latentes. DLGMs forman una familia unificada de modelos que incluyen análisis factorial [Bar11], análisis factorial no lineal [LH00], y redes de creencias Gaussianas no-lineales (en inglés, *non-linear Gaussian belief networks*) [FH99]. Otros modelos relacionados incluyen redes de creencias sigmoides (sigmoid belief) [SJJ96] y deep autoregressive networks [Gre+14], las cuales utilizan distribuciones de Bernoulli auto regresivas en cada capa en lugar de distribuciones gaussianas. DLGMs son un método de inferencia de propósito general para modelos con variables latentes continuas. El proceso de los DLGMs se puede observar en la figura (8).

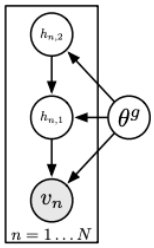


Figure 8: Modelo gráfico para DLGMs

En [Lia+18] consideran un modelo generativo similar a los DLGMs. Considerando a $u \in \{1, ..., U\}$ para indexar a los usuarios y a $i \in \{1, ..., I\}$ para indexar a los ítems. Sea $X \in \mathbb{R}^{U \times I}$ la matriz de interacción entre los usuarios y los ítems. Para cada usuario u el modelo muestrea la representación latente z_u $K-$dimensional de una priori Gaussiana estándar. La representación latente z_u es transformada mediante una función no lineal $f_\theta(\cdot) \in \mathbb{R}^I$ para producir una distribución de probabilidad $\pi(z_u)$ sobre los ítems I a partir de ellos se extrae x_u de la siguiente manera:

$$
\begin{aligned}
\mathrm{z}_u &\sim \mathcal{N}(0, \mathrm{I}_K) \\
\boldsymbol{\pi}(\mathrm{z}_u) &\propto \exp\{f_\theta(\mathrm{z}_u)\} \\
\mathrm{x}_u &\sim \mathrm{Mult}(N_u, \boldsymbol{\pi}(\mathrm{z}_u))
\end{aligned}
$$

La función no lineal $f_\theta(\cdot)$ es un perceptrón multicapa con parámetros θ. Dado el número total de interacciones $N_u = \sum_i x_{ui}$ del usuario u, se muestrea x_u a partir de una distribución multinomial con probabilidad $\pi(z_u)$.

La log-verosimilitud para el usuario u, condicionada a la representación latente, es

$$
\log p_\theta(\mathrm{x}_u \mid \mathrm{z}_u) \stackrel{c}{=} \sum_i x_{ui} \log \pi_i(\mathrm{z}_u). \tag{10}
$$

Esta verosimilitud pone la probabilidad de masa en las entradas distintas de cero en x_u. El modelo debe asignar más probabilidad de masa a los ítems que tienen más probabilidad de interacción ya que se tiene probabilidad de masa limitada porque $\pi(z_u)$ debe sumar 1.

9.2 Inferencia Variacional

La idea principal de los métodos variacionales es considerar la inferencia como un problema de optimización. Suponiendo que se tiene una distribución de probabilidad a posterior *intratable*. Las técnicas variacionales intentarán resolver un problema de optimización sobre una clase de distribuciones controlables Q para encontrar un $q \in Q$ que sea más simple y similar a la probabilidad a posterior. Luego, se considera la distribución variacional $q(z_u)$ (en lugar de la posterior)

para obtener una solución aproximada.

Las principales diferencias entre el muestreo y las técnicas variacionales son:

- A diferencia de los métodos basados en muestreo, los enfoques variacionales casi nunca encontrarán la solución globalmente óptima
- Sin embargo, siempre se sabrá si han convergido.
- En la práctica, los métodos de inferencia variacional a menudo se escalan mejor y se adaptan mejor a técnicas como la optimización de gradiente estocástico, la paralelización en varios procesadores y la aceleración mediante GPU.

Se define $q(z_u)$ como una distribución (variacional) Gaussiana completamente factorizada (diagonal):

$$q(\mathrm{z}_u) = \mathcal{N}\left(\mu_u, \mathrm{diag}\left\{\sigma_u^2\right\}\right).$$

El objetivo de la inferencia variacional es optimizar los parámetros variacionales libres $\left\{\mu_u, \sigma_u^2\right\}$ de tal manera que la *divergencia de Kullback-Leiber* $\mathrm{KL}\left(q(\mathrm{z}_u) \| p(\mathrm{z}_u \mid \mathrm{x}_u)\right)$ sea mínima.

La divergencia de Kullback-Leiber es una medida no simétrica de la similitud o diferencia entre dos funciones de distribución de probabilidad. Para distribuciones $q(\mathrm{z}_u)$ y $p(\mathrm{z}_u \mid \mathrm{x}_u)$ de una variable aleatoria continua, la divergencia KL se define como la integral:

$$\mathrm{KL}\left(q(\mathrm{z}_u) \| p(\mathrm{z}_u \mid \mathrm{x}_u)\right) = \int_{-\infty}^{\infty} q(\mathrm{z}_u) \ln \frac{q(\mathrm{z}_u)}{p(\mathrm{z}_u \mid \mathrm{x}_u)} \mathrm{d}z_u.$$

Con la inferencia variacional, el número de parámetros a optimizar $\{\mu_u, \sigma_u^2\}$ crece con el número de usuarios y elementos en el conjunto de datos. Los **variational autoencoders** (**VAEs**) reemplazan los parámetros variacionales individuales con una función que depende de los datos:

$$g_\phi(\mathrm{x}_u) \equiv \left[\mu_\phi(\mathrm{x}_u), \sigma_\phi(\mathrm{x}_u)\right] \in \mathbb{R}^{2K}$$

parametrizado por ϕ con K-vectores $\mu_\phi(\mathrm{x}_u)$ y $\sigma_\phi(\mathrm{x}_u)$. Entonces, la distribución variacional es

$$q_\phi(\mathrm{z}_u \mid \mathrm{x}_u) = \mathcal{N}\left(\mu_\phi(\mathrm{x}_u), \mathrm{diag}\left\{\sigma_\phi^2(\mathrm{x}_u)\right\}\right).$$

Es decir, usando los datos observados x_u, el modelo de inferencia genera los correspondientes parámetros variacionales de la distribución variacional $q_\phi(z_u|x_u)$, que, cuando se optimiza, se aproxima a la posterior $p(z_u|x_u)$. Poniendo juntos a $q_\phi(\mathrm{z}_u \mid \mathrm{x}_u)$ y al modelo generativo $p_\theta(\mathrm{x}_u \mid \mathrm{z}_u)$ en la figura 9, se obtiene una arquitectura neuronal que se asemeja a un autoencoder.

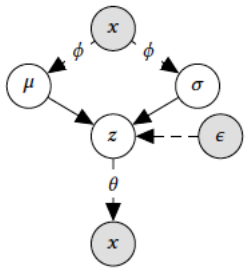


Figure 9: Taxonomía de un Variational Autoencoder

Los VAEs analizan las preferencias del usuario mediante la explotación de los patrones de similitud inferidos a partir de experiencias pasadas.

Para el entrenamiento de los VAEs se acota inferiormente a la log-verosimilitud marginal de los datos, comunmente conocido como *evidence lower bound* (ELBO). Para el usuario u se busca maximizar

$$\log p\left(\mathrm{x}_u;\theta\right) \geq \mathbb{E}_{q_\phi\left(\mathrm{z}_u|\mathrm{x}_u\right)}\left[\log p_\theta\left(\mathrm{x}_u \mid \mathrm{z}_u\right)\right] - \mathrm{KL}\left(q_\phi\left(\mathrm{z}_u \mid \mathrm{x}_u\right) \| p\left(\mathrm{z}_u\right)\right) \equiv \mathcal{L}\left(\mathrm{x}_u;\theta,\phi\right). \tag{11}$$

Se puede conseguir una estimación insesgada de ELBO muestreamndo $z_u \sim q_\phi$ y realizando gradiente ascendiente estocástico para optimizarlo. Para esto, se considera la siguiente *reparametrización*: $z_u = \mu_\phi(x_u) + \varepsilon \odot \sigma_\phi(x_u)$ con $\varepsilon \sim \mathcal{N}(0, I_K)$. Al hacerlo, la estocasticidad en el proceso de muestreo se aísla y el gradiente con respecto a ϕ se puede propagar hacia atrás a través del z_u muestreado.

Se puede interpretar el primer término de ELBO (11) como un error de reconstrucción y al segundo término como un término de reparametrización. Bajo esta perspectiva se introduce un parámetro β que controla la regularización

$$\mathcal{L}_\beta\left(\mathrm{x}_u;\theta,\phi\right) \equiv \mathbb{E}_{q_\phi\left(\mathrm{z}_u|\mathrm{x}_u\right)}\left[\log p_\theta\left(\mathrm{x}_u \mid \mathrm{z}_u\right)\right] - \beta \cdot \mathrm{KL}\left(q_\phi\left(\mathrm{z}_u \mid \mathrm{x}_u\right) \| p\left(\mathrm{z}_u\right)\right) \tag{12}$$

El punto de vista de regularización de ELBO introduce una compensación entre qué tan bien podemos ajustar los datos y qué tan cerca permanece el posterior aproximado del priori durante el aprendizaje. Se comienza el entrenamiento con $\beta = 0$ y gradualmente se aumenta β a 1.

9.3 Taxonomía de Autoencoders (AE)

Desde la perspectiva del aprendizaje de autoencoders, la estimación de máxima verosimilitud en un autoencoder regular toma la siguiente forma

$$\begin{aligned}\theta^{\mathrm{AE}}, \phi^{\mathrm{AE}} &= \arg\max_{\theta,\phi} \sum_u \mathbb{E}_{\delta\left(\mathrm{z}_u - g_\phi(\mathbf{x}_u)\right)}\left[\log p_\theta\left(\mathrm{x}_u \mid \mathrm{z}_u\right)\right] \\ &= \arg\max_{\theta,\phi} \sum_u \log p_\theta\left(\mathrm{x}_u \mid g_\phi\left(\mathrm{x}_u\right)\right)\end{aligned}.$$

Hay que notar 2 aspectos importantes:

- Autoencoders (y denoising autoencoders) optimizan el primer término en el VAE objetivo (ecuaciones 11 y 12) usando una distribución variacional delta $q_\phi\left(\mathrm{z}_u \mid \mathrm{x}_u\right) = \delta\left(\mathrm{z}_u - g_\phi\left(\mathrm{x}_u\right)\right)$; no regularizan $q_\phi\left(\mathrm{z}_u \mid \mathrm{x}_u\right)$ con cualquier distribución a priori como los VAE.
- El $\delta\left(\mathrm{z}_u - g_\phi\left(\mathrm{x}_u\right)\right)$ es una distribución δ con masa en la salida de $g_\phi\left(\mathrm{x}_u\right)$. Al contrario que con los VAE, donde el aprendizaje se realiza mediante una distribución variacional, es decir, $g_\phi(x_u)$ genera los parámetros (media y varianza) de una distribución Gaussiana. Es decir, los VAE tienen la capacidad de capturar variaciones por punto de datos en el estado latente z_u.

En la Figura 10a se puede observar la taxonomía de un autoencoder. Agregar ruido a la entrada (o la representación oculta intermedia) de un autoencoder se obtiene un denoising autoencoder (Figura 10b).

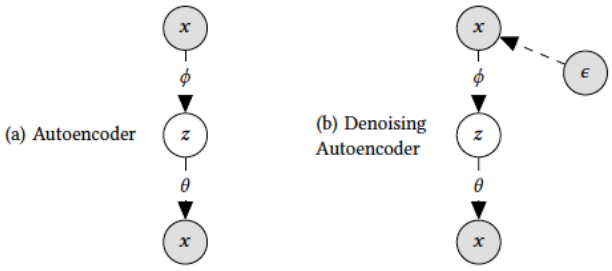


Figure 10: Taxonomía de Autoencoders

Bibliografía

[Dev86] Luc Devroye. "Sample-Based Non-Uniform Random Variate Generation". In: *Proceedings of the 18th Conference on Winter Simulation*. WSC '86. Washington, D.C., USA: Association for Computing Machinery, 1986, pp. 260–265. ISBN: 0911801111. DOI: 10.1145/318242.318443. URL: https://doi.org/10.1145/318242.318443.

[Gol+92] David Goldberg et al. "Using collaborative filtering to weave an information tapestry". In: *Communications of the ACM* 35.12 (1992), pp. 61–70.

[Fau94] Laurene Fausett. *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*. USA: Prentice-Hall, Inc., 1994. ISBN: 0133341860.

[BDH96] C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa. "The Quickhull Algorithm for Convex Hulls". In: *ACM Trans. Math. Softw.* 22.4 (Dec. 1996), pp. 469–483. ISSN: 0098-3500. DOI: 10.1145/235815.235821. URL: https://doi.org/10.1145/235815.235821.

[SJJ96] L. K. Saul, T. Jaakkola, and M. I. Jordan. "Mean Field Theory for Sigmoid Belief Networks". In: *Journal of Artificial Intelligence Research* 4 (1996), pp. 61–76. DOI: 10.1613/jair.251.

[Par98] Giorgio Parisi. *Statistical field theory*. Perseus Books, 1998.

[FH99] Brendan J. Frey and Geoffrey E. Hinton. "Variational Learning in Nonlinear Gaussian Belief Networks". In: *Neural Computation* 11.1 (1999), pp. 193–213. DOI: 10.1162/089976699300016872.

[Hay99] Simon Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1999.

[Her+99] Jonathan L Herlocker et al. "An algorithmic framework for performing collaborative filtering". In: *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. 1999, pp. 230–237.

[CS00] Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000. DOI: 10.1017/CBO9780511801389.

[LH00] Harri Lappalainen and Antti Honkela. "Bayesian Non-Linear Independent Component Analysis by Multi-Layer Perceptrons". In: *Advances in Independent Component Analysis Perspectives in Neural Computing* (2000), pp. 93–121. DOI: 10.1007/978-1-4471-0443-8_6.

[Sar+01] Badrul Sarwar et al. "Item-based collaborative filtering recommendation algorithms". In: *Proceedings of the 10th international conference on World Wide Web*. 2001, pp. 285–295.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 0387310738.

[Yan07] Xitong Yang. "Understanding the Variational Lower Bound". In: *Institute for Advanced Computer Studies* (2007).

[Bar11] David Bartholomew. *Latent Variable Models and Factor Analysis. A Unified Approach*. 3. Auflage. Chichester: Wiley, 2011. ISBN: 9780470971925.

[Lee+13] Joonseok Lee et al. "Local low-rank matrix approximation". In: *International conference on machine learning*. PMLR. 2013, pp. 82–90.

[Gre+14] Karol Gregor et al. "Deep AutoRegressive Networks". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1242–1250. URL: https://proceedings.mlr.press/v32/gregor14.html.

[KW14] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2014. arXiv: 1312.6114 [stat.ML].

[RMW14] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1278–1286. URL: https://proceedings.mlr.press/v32/rezende14.html.

[Che+16] Heng-Tze Cheng et al. “Wide & deep learning for recommender systems”. In: *Proceedings of the 1st workshop on deep learning for recommender systems*. 2016, pp. 7–10.

[CK16] Evangelia Christakopoulou and George Karypis. “Local item-item models for top-n recommendation”. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. 2016, pp. 67–74.

[CAS16] Paul Covington, Jay Adams, and Emre Sargin. “Deep neural networks for youtube recommendations”. In: *Proceedings of the 10th ACM conference on recommender systems*. 2016, pp. 191–198.

[Lee+16] Joonseok Lee et al. “LLORMA: Local low-rank matrix approximation”. In: (2016).

[Beu+17] Alex Beutel et al. “Beyond globally optimal: Focused learning for improved recommendations”. In: *Proceedings of the 26th International Conference on World Wide Web*. 2017, pp. 203–212.

[CK18] Evangelia Christakopoulou and George Karypis. “Local latent space models for top-n recommendation”. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2018, pp. 1235–1243.

[He+18] Xiangnan He et al. “Outer product-based neural collaborative filtering”. In: *arXiv preprint arXiv:1808.03912* (2018).

[Lia+18] Dawen Liang et al. *Variational Autoencoders for Collaborative Filtering*. 2018. arXiv: 1802.05814 [stat.ML].

[DCJ19] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. “Are we really making much progress? A worrying analysis of recent neural recommendation approaches”. In: *Proceedings of the 13th ACM Conference on Recommender Systems*. 2019, pp. 101–109.

[Lud+19] Malte Ludewig et al. “Performance comparison of neural and non-neural approaches to session-based recommendation”. In: *Proceedings of the 13th ACM conference on recommender systems*. 2019, pp. 462–466.

[Cho+21] Minjin Choi et al. “Local Collaborative Autoencoders”. In: *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 2021, pp. 734–742.

[Doe21] Carl Doersch. *Tutorial on Variational Autoencoders*. 2021. arXiv: 1606.05908 [stat.ML].