UNIVERSIDAD
DE LA REPUBLICA
URUGUAY

LOGO 1 | LOGO 2 | LOGO 3

Aprendizaje Profundo utilizando Modelos de Difusión

Modelos de Difusión

Nombres del autor José María Clavijo Burgueño

Programa de Posgrado en Ingeniería Matemática
Facultad de Ingeniería
Universidad de la República

Montevideo – Uruguay
Agosto de 2024

UNIVERSIDAD
DE LA REPUBLICA
URUGUAY

LOGO 1 | LOGO 2 | LOGO 3

Aprendizaje Profundo utilizando Modelos de Difusión

Modelos de Difusión

Nombres del autor José María Clavijo Burgueño

Tesis de Maestría presentada al Programa de Posgrado en Ingeniería Matemática, Facultad de Ingeniería de la Universidad de la República, como parte de los requisitos necesarios para la obtención del título de Magister en Ingeniería Matemática.

Director de tesis:
Ph D. Prof. Ernesto Mordecki Apellido

Director académico:
Ph D. Prof. Ernesto Mordecki Apellido

Montevideo – Uruguay
Agosto de 2024

José María Clavijo Burgueño, Nombres del autor

Aprendizaje Profundo utilizando Modelos de Difusión / Nombres del autor José María Clavijo Burgueño. - Montevideo: Universidad de la República, Facultad de Ingeniería, 2024.

xii, 38 p. 29, 7cm.

Director de tesis:

Ernesto Mordecki Apellido

Director académico:

Ernesto Mordecki Apellido

Tesis de Maestría – Universidad de la República, Programa de Ingeniería Matemática, 2024.

Referencias bibliográficas: p. 21 – 21.

1. Difusión, 2. Denoising, 3. Aprendizaje, 4. Profundo, 5. Modelos. I. Apellido, Ernesto Mordecki. II. Universidad de la República, Programa de Posgrado en Ingeniería Matemática. III. Título.

INTEGRANTES DEL TRIBUNAL DE DEFENSA DE TESIS

_____

D.Sc. Prof. Nombre del 1er Examinador Apellido

_____

Ph.D. Prof. Nombre del 2do Examinador Apellido

_____

D.Sc. Prof. Nombre del 3er Examinador Apellido

_____

Ph.D. Prof. Nombre del 4to Examinador Apellido

_____

Ph.D. Prof. Nombre del 5to Examinador Apellido

Montevideo – Uruguay
Agosto de 2024

(Dedicatoria) A alguien cuyo
valor es digno de ella.

Agradecimentos

Quisiera agredecer a...

(Epígrafe:) *Frase que alude al tema de trabajo.*

Autor

RESUMEN

En esta tesis se presenta...

Palabras claves:
Difusión, Denoising, Aprendizaje, Profundo, Modelos.

ABSTRACT

In this work, we present ...

Keywords:
Difusión, Denoising, Aprendizaje, Profundo, Modelos.

Lista de símbolos

Lista de los símbolos más relevantes de la tesis.

α Escalar 18

$\emptyset$ Conjunto que carece de elementos 18

$\mathbb{R}$ Conjunto de los números reales 18

$\boldsymbol{\sigma}$ Tensor simétrico de tensiones de Cauchy. 18

Tabla de contenidos

Capítulo 1

Introducción

El presente trabajo tiene como referencia motivadora el artículo **"Deep Unsupervised Learning using Nonequilibrium Thermodynamics"**Jascha Sohl-Dickstein (2015).

La idea esencial, inspirada en el no equilibrio física estadística, es destruir la estructura en una distribución de datos a través de un proceso iterativo de difusión hacia adelante. Luego de eso, se realizará un proceso de difusión inversa que restaura la estructura de los datos, lo que da lugar a una estructura altamente flexible y un modelo generativo manejable de los datos. Este enfoque nos permite aprender, muestrear y evaluar las probabilidades en los modelos generativos profundos con miles de capas o pasos de tiempo, así como para calcular probabilidades condicionales y a posteriori bajo el modelo aprendido.

Estos Modelos de Difusión están basados en que es difícil convertir ruido a datos estructurados, pero es fácil convertir datos estructurados en ruido. La técnica nos dice que podemos usar un **proceso hacia adelante** o **proceso de difusión** a los efectos de convertir los datos observados x_0 en otra versión x_T pasando los datos por un procesamiento de T pasos de un codificador estocástico $q(x_t|x_{t-1})$. Después de una cantidad suficiente de pasos, tendremos $x_T \sim \mathcal{N}(0,\, I)$ (una disribución Gausiana Estándar), o alguna otra distribución conveniente.

Entonces, aprenderemos el proceso inverso para deshacer este proceso, pasando el ruido obtenido a través de T pasos por un decodificador $p_\theta(x_{t-1}|x_t)$ hasta

llegar a x_0. Aprender el proceso inverso significa aplicar una Red Neuronal para aprender los parámetros θ de dicha red.

1.1. Proceso "hacia delante" - Codificador

El primer proceso es el proceso hacia delante, que también nos podemos encontrar con la nomenclatura "q", prior, o forward noising process (en inglés).Su objetivo es generar los datos o muestras de entrenamiento.

Para ello, a partir de una imagen inicial x_0, se añade ruido en pequeños pasos a lo largo de una cadena de Markov, es decir, cada estado en un momento de tiempo depende únicamente del anterior $q(x_{t-1}|x_t)$. Las muestras se construyen paso a paso, tomando una serie de supuestos derivados de las propiedades de matrices y del teorema central del límite:

- El ruido añadido es una gaussiana donde las variables son independientes. O, lo que es lo mismo, cada muestra generada es independiente del anterior: $q(x_{t-1}|x_t)$ es independiente de $q(x_{t-2}|x_{t-1})$.
- Si se añaden suficientes pasos (en la publicación original eran 1.000, T=1000), la distribución de la muestra final x_T es una gaussiana.

En resumen: las muestras de los datos son independientes entre sí y al final es solo ruido. Estos supuestos son lo que facilitan la estimación del proceso reverso de reconstrucción de la imagen se verá más adelante.

1.2. Proceso reverso - Decodificación

El segundo proceso es el proceso reverso o "hacia atrás", que también podemos encontrar como p_θ, posterior o reverse diffusion process (en inglés). Su fin es generar la imagen objetivo gradualmente a partir del ruido.
Para ello, se entrena una red neuronal que aprende a generar la imagen objetivo a partir de los datos de entrenamiento generados en el proceso "hacia delante". Es a partir de aquí donde los supuestos con los que se ha generado la muestra de entrenamiento facilitan el aprendizaje:

- Por un lado, se realiza paso a paso a través de otra cadena de Markov, de nuevo, mediante pasos de estado temporales independientes.

- Por otro lado, como el ruido final generado en el primer proceso x_T es una normal o gaussiana, podemos generar la imagen original x_0 a partir del proceso para la distribución inversa $q(x_{t-1}|x_t)$. Recordar, que la q se refieres al proceso hacia adelante.
- Finalmente, como no conocemos toda la distribución inversa, la cadena de Markov en muchos pasos nos permite aproximar, de nuevo, a una gaussiana que llamamos p_θ.
- El objetivo del entrenamiento consiste en que la distribución de $p_\theta(t-1|t)$ (aproximada o estimada) y $q(x_{t-1}|x_t)$ (real), sean los más parecidas o en notación matemática, que el "variational lower bound" (la distancia) entre p y q sea mínima.

Con esto, ya tenemos todos lo ingredientes para resolver nuestra ecuación que consiste en encontrar la media y la matriz de covarianza de la aproximación gaussiana p_θ en cada paso. Esta media y covarianza serán los parámetros que, una vez entrenado, se utilizarán para la inferencia en el momento de la predicción.

El proceso se puede simplificar prediciendo el ruido que hay que sustraer en cada paso, en lugar de predecir la imagen que hay que generar, pero aquí, que cada científico utilice la mejor aproximación.

Capítulo 2

Fundamentos Teóricos - Modelos de Difusión

En esta parte, veremos con mayor detalle cómo se elimina el ruido en los modelos probabilísticos de difusión. Se exploran las bases matemáticas sobre la cuál están basados estos procesos de codificación y decodificación. Nos hemos asado en la siguiente fuente bibliográfica Murphy (2023) para esta parte.

2.1. Difusión hacia Adelante - Codificador

El codificador **hacia adelante** está definido como un **modelo lineal Gausiano**: $q(x_t|x_{t-1}) = \mathcal{N}(x_t|\sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I})$ dónde β_t es la varianza en el momento t, $\beta_t \in (0,1)$, y dichos valores serán elegidos para asegurarnos que obtengamos en el siguiente paso una distribución normal.

Podemos reformular lo anterior con la fórmula de la distribución normal: $\mathcal{N}(\mu, \Sigma^2) = \mu + \Sigma.\epsilon$, $\epsilon = \mathcal{N}(0, \mathbf{I})$ con media $\mu = \sqrt{1-\beta_t}x_{t-1}$ y varianza $\Sigma^2 = \beta_t\mathbf{I}$.

El parámetro β_t lo usaremos para añadir el ruido en cada paso. Esto se hace progresivamente, por tanto no es un valor fijo; en cada paso tendremos un β específico.
Podemos imaginar los β_t como una lista de valores que va guardando cómo cambian los valores de para cada instante de tiempo o paso. Y cuando necesitamos saber el valor de en un momento específico, simplemente lo referenciamos con

el índice t o el índice del paso. Para implementar esto, por ejemplo en Python, se utiliza un Scheduler o planificador de ruido.
Con lo anterior , podemos escribirla de la siguiente manera:

$$q(x_t|x_{t-1}) = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}.\epsilon \tag{2.1}$$

.
Entonces para llegar a cualquier momento T, se necesita aplicarla de manera iterativa desde $t = 0$ y se puede formalizar de la siguiente forma :

$$q(x_{1:T}|x_0) := \prod_{t=1}^{T} q(x_t|x_{t-1}) \tag{2.2}$$

.
Observando esta fórmula no seríamos computacionalmente eficientes si la aplicáramos iterativamente ,considerando que durante el entrenamiento necesitamos evaluar nuestra red con diferentes valores de t. Entonces conviene tomar un atajo que consiste en no tener que recorrer toda la cadena hasta llegar a cada paso específico de t, dicho de otra forma no será necesario pasar por todos los pasos intermedios , entre x_0 y x_t, para llegar a un t específico. Se replantea la fórmula anterior y para ellos generamos nuevas variables:

$$\alpha_t = 1 - \beta_t \tag{2.3}$$

$$\overline{\alpha_t} = \prod_{s=1}^{t} \alpha_t \tag{2.4}$$

Entonces, $\overline{\alpha_t}$, contendrá la acumulación de todas las β anteriores en la cadena. Esto nos permite reformular la Fórmula 2.1 de una manera más eficiente y directa, facilitando el salto desde x_0 hasta x_t:

$$q(x_t|x_0) = \sqrt{\overline{\alpha_t}}x_0 + \sqrt{1-\overline{\alpha_t}}\epsilon \tag{2.5}$$

Ahora usando la fórmula 2.5, contamos con un mecanismo eficiente para transformar cualquier muestra original x_0 en su versión ruidosa x_t, utilizando un único salto hacia adelante. Cuando hablemos del proceso hacia adelante o forward, nos estaremos refiriendo específicamente a $q(x_t|x_0)$ según lo define la Fórmula 2.5.
Vamos a realizar la aplicación del ruido tal que $\overline{\alpha_T} \approx 0$, de forma que

$q(x_T|x_0) \approx \mathcal{N}(0, \mathbf{I})$. La distribución $q(x_t|x_0)$ se conoce como el **kernel de difusión**.

2.2. Scheduler - Planificador del Ruido

La cantidad de ruido que añadimos en cada etapa de nuestra cadena de Markov se diseña cuidadosamente para asegurar que, al final del proceso, terminemos $x_T \sim \mathcal{N}(0, \mathbf{I})$.

Si optáramos por agregar una cantidad constante de ruido en cada paso, la variabilidad de nuestra distribución crecería mucho debido a la acumulación de este ruido. Entonces, se torma muy importante ajustar el incremento del ruido que añadimos paso a paso. Aquí es donde se usa el Scheduler y el β_t, que determina cuánto ruido se agregaría en cada paso.

Al inicio de cada paso generamos un nuevo conjunto de valores ruido que se obtienen de $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, y ajustamos su intensidad usando β_t. Entonces la cantidad de ruido que insertamos varía en casa paso.

2.3. Difusión hacia Atrás - Decodificador - Eliminación de Ruido

Necesitamos ahora poder revertir el proceso, o sea, cómo retroceder a través de la cadena de Markov para recuperar la muestra original x_{t-1}, partiendo de su estado corrupto x_t, queremos obtener $q(x_{t-1}|x_t, x_0)$. En el proceso inverso de difusión buscamos, con x_0 conocido, podemos derivar el inverso de uno de los pasos hacia adelante de la siguiente manera:

$$q(x_t|x_{t-1}, x_0) = \frac{\overbrace{q(x_{t-1}|x_t, x_0)}^{\text{posterior}} q(x_t|x_0)}{q(x_{t-1}|x_0)} \tag{2.6}$$

Se aplica el teorema de Bayes, (ver Fórmula 2.6), para conseguir derivar este factor del numerador $q(x_{t-1}|x_t, x_0)$ denominado "posterior". Ver [1] para ver detalles de esta derivación; también para esta sección se consultó el si-

[1] https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

guiente artículo [2]. Este nos define cómo obtener el paso previo en la cadena, condicionado por el paso actual y por la muestra original. La definición de este posterior es la siguiente:

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I) \tag{2.7}$$

$$\tilde{\mu}_t(x_t, x_0) := \frac{\sqrt{\overline{\alpha}_{t-1}}\beta_t}{1-\overline{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}x_t \tag{2.8}$$

$$\tilde{\beta}_t := \frac{1-\overline{\alpha}_{t-1}}{1-\overline{\alpha}_t}\beta_t \tag{2.9}$$

La media condicional $\tilde{\mu}_t(x_t, x_0)$ se obtiene combinando x_t y x_0 ponderados por sus respectivas contribuciones de ruido. La derivación de esta media condicional se basa en la propiedad de las distribuciones normales y en cómo se combinan las varianzas y medias en el proceso de difusión. Aquí está el desglose de los términos:

$\sqrt{\overline{\alpha}_{t-1}}$ y $\sqrt{\alpha_t}$ son factores que ajustan la escala de x_0 y x_t respectivamente. β_t es el parámetro de ruido en el paso t. $\overline{\alpha}_t$ es el producto acumulativo de α hasta el tiempo t. La combinación de estos términos asegura que la media condicional $\tilde{\mu}_t(x_t, x_0)$ sea una estimación óptima para revertir el ruido añadido en el proceso de difusión.

La varianza $\tilde{\beta}_t$ se deriva considerando cómo se propaga el ruido a través del proceso de difusión. La relación entre β_t y $\tilde{\beta}_t$ se obtiene al ajustar la varianza acumulada del ruido en cada paso. La fórmula para $\tilde{\beta}_t$ asegura que la varianza en el proceso inverso se ajuste correctamente para reflejar la incertidumbre residual en cada paso de la difusión.

El cociente de la Fórmula (2.9) ajusta la varianza del ruido en el proceso inverso de difusión. Vamos a desglosar su significado:

- $\overline{\alpha}_t$: Es el producto acumulativo de α hasta el tiempo (t), es decir, $\overline{\alpha}t = \prod i = 1^t \alpha_i$. Representa la cantidad de información original que se conserva en el paso (t).
- $\overline{\alpha}_{t-1}$: Similarmente, $\overline{\alpha}_{t-1}$ es el producto acumulativo de α hasta el tiempo $t-1$.

[2] https://javiersolisgarcia.com/posts/ddpm/

- Cociente $\frac{1-\overline{\alpha}_{t-1}}{1-\overline{\alpha}_t}$: Este cociente ajusta la varianza del ruido β_t para reflejar la proporción de ruido que se ha añadido entre los pasos (t-1) y (t).

Su interpretación es la siguiente:

- Numerador $1-\overline{\alpha}_{t-1}$: Representa la cantidad de ruido acumulado hasta el paso (t-1).
- Denominador $1-\overline{\alpha}_t$: Representa la cantidad de ruido acumulado hasta el paso (t).
- El cociente $\frac{1-\overline{\alpha}_{t-1}}{1-\overline{\alpha}_t}$ ajusta (β_t) para que la varianza en el paso (t) refleje correctamente la cantidad de ruido acumulado en el proceso de difusión. Este ajuste es crucial para que la varianza estimada $\tilde{\beta}_t$ sea precisa y permita una correcta reversión del proceso de difusión.

Es cierto que depender de x_0 en las fórmulas, especialmente las Fórmulas 2.7 y 2.8 , no es conveniente ya que cuando necesitemos generar nuevas muestras, no tendremos disponible a x_0. Por lo tanto, para eliminar la dependencia de x_0, establecemos que $q(x_t|x_0) = x_t$ lo que constituye un "truco"práctico. Podemos despejar de la fórmula (2.5) a x_0 y tenemos :

$$x_0 = \frac{1}{\sqrt{\overline{\alpha}_t}}(x_t - \frac{1-\alpha}{\sqrt{1-\overline{\alpha}_t}}\epsilon) \tag{2.10}$$

y luego, lo obtenido lo aplicamos a la fórmula (2.8) obteniendo:

$$\tilde{\mu}(x_t, x_0) = \frac{1}{\sqrt{\alpha}}(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon) \tag{2.11}$$

Para ir terminando el proceso hacia atrás se realizan a continuación unos ajustes a la Fórmula (2.7). Veamos a continuación:

- Se aplica que $\mathcal{N}(\mu, \Sigma^2) = \mu + \Sigma\epsilon$ con $\epsilon \sim \mathcal{N}(0, I)$.
- Aplicar la definición de $\tilde{\mu}(x_t, x_0)$ obtenido en la Fórmula (2.11).
- Sabemos que $x_t = q(x_{t-1}|x_t, x_0)$. Esta expresión nos permite modelar el proceso inverso de manera que podamos revertir el ruido añadido en cada paso y recuperar la imagen original x_0.

Se llega entonces a la siguiente fórmula :

$$x_{t-1} = \frac{1}{\sqrt{\alpha}}(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}}}\epsilon_t) + \sqrt{\beta_t}\epsilon \tag{2.12}$$

Observando la fórmula tenemos que distinguir entre dos tipos de **epsilon**, ϵ_t y ϵ. ¿ Qué rol cumple cada uno de estos ?:

- ϵ representa el ruido que se introduce al aplicar la normal. Este es el ruido específico que se añade en un paso dado para ajustarse a la definición de la distribución normal que aplicamos al volver hacia atrás.
- ϵ_t Representa la suma total del ruido aplicado a x_0 para transformarlo en x_t. Este valor resume la acumulación de ruido a lo largo de los pasos hasta alcanzar el estado actual.

2.4. Proceso hacia atrás $p_\theta(x_{t-1}|x_t)$.

En este punto ya sabemos como ir hacia adelante y hacia atrás por la cadena de Markov, entonces el objetivo principal se centra en capacitar a nuestro modelo para que realice este proceso inverso eficazmente. Buscamos que $p_\theta(x_{t-1}|x_t)$ haga la misma tarea que $q(x_{t-1}|x_t, x_0)$.

Acá el θ representa a los parámetros de la distribución que permite obtener la imagen x_0, y para ello necesitamos usar una red neuronal para lograrlo.

La definición del proceso inverso se establece de la siguiente manera:

$$p(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{2.13}$$

Esta fórmula es análogo a la Fórmula (2.7), nuestro objetivo es ser capaces de predecir la distribución de la muestra en el paso anterior. Mirando la fórmula antarior, deberíamos contar con dos modelos diferentes que puedan predecir tanto la media $\mu_\theta(x_t, t)$ como la varianza $\Sigma_\theta(x_t, t)$ de la distribución normal en cuestión.

Pero, los autores del artículo original descubrieron que omitir la predicción de $\Sigma_\theta(x_t, t)$ llevaba a resultados más eficientes. Dado que la varianza ya está determinada por el Scheduler, podemos prescindir de predecirla, lo cual simplifica el proceso. También hay investigaciones posteriores han encontrado ventajas en predecir $\Sigma_\theta(x_t, t)$.

Entonces la formulación del proceso inverso se simplifica considerablemente:

$$p(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \beta_t I) \tag{2.14}$$

La media que tenemos que predecir entonces es la siguiente :

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha_t}}}\epsilon_\theta(x_t, t)) \tag{2.15}$$

Entonces lo primordial a predecir es $\epsilon_\theta(x_t, t)$, el cual representa efectivamente el ruido que se ha ido acumulando durante toda la cadena hasta el punto . Esto nos indica cómo se ha modificado la muestra original para llegar a este estado específico de este paso de la cadena. Al predecir correctamente este ruido, podemos ajustarlo para el paso actual con el factor $\frac{\beta_t}{\sqrt{1-\overline{\alpha_t}}}$ y con esto se va a ir eliminando gradualmente al ir hacia atrás en la cadena.

Según todo lo visto hasta acá, llegamos a la ecuación definitiva que nos permite retroceder paso a paso a lo largo de la cadena de Markov. Este proceso nos muestra cómo, a través de la predicción precisa del ruido en cada etapa, podemos retroceder por el camino de la difusión para recuperar la muestra original desde su estado corrupto.

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha_t}}}\epsilon_\theta(x_t, t)) + \sqrt{\beta_t}\epsilon \tag{2.16}$$

Una vez más, vuelve a aparecer dos **epsilon** distintos, los volvemos a definir:

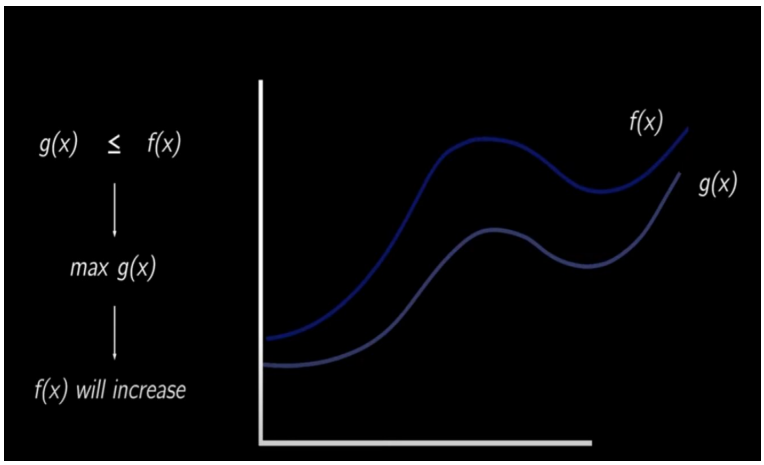
- ϵ es el ruido que se le aplica para cumplir la definición de la normal.
- $\epsilon_\theta(x_t, t)$ es el valor de ruido predicho por el modelo para calcular la media. Este valor representa el ruido acumulado que ha sido aplicado hasta el punto en la cadena. Predecir este ruido con precisión es esencial para poder revertir el proceso de difusión, eliminando el ruido añadido y recuperando la muestra original a medida que retrocedemos a través de la cadena.

2.4.1. Model Fitting - Ajuste del Modelo y Función de Pérdidas.

Para que nuestro modelo pueda encontrar los parámetros θ que nos permitan deshacer el ruido con precisión, necesitamos una función de pérdida o función objetivo. En primera instancia, se pretende que esta función sea la log-likehood de $p_\theta(x_0)$, lo cual implicaría recorrer toda la cadena para cada batch [3] de entrenamiento. Esto conforma un proceso extremadamente costoso y prácticamente inviable en términos de tiempo de computación a emplear, lo que señala el artículo de referencia definiéndolo como un proceso intratable (intractable).

Dicho lo anterior recurrimos al concepto del **Variational Lower Bound** también denominada **Evidende Lower Bound (ELBO)**, que siempre es una **cota inferior** para la log-likehood. Al optimizar esta función, indirectamente estamos mejorando la log-likehood, ya que el Variational Lower Bound siempre se sitúa por debajo de esta última.

Esta relación se ilustra claramente con la siguiente imagen.[4].



La relación que se muestra en el gráfico se formaliza de la siguiente manera :

[3]Un batch es un subconjunto del conjunto de datos de entrenamiento. En lugar de procesar todos los datos de una vez, el modelo se entrena en pequeños lotes de datos.

[4]Imagen tomada de https://javiersolisgarcia.com/posts/ddpm

$$\mathbb{E}[\underbrace{-\log p_\theta(x_0)}_{NLL}] \leq \mathbb{E}_q[\underbrace{-\log p_\theta(x_0|x_1)}_{L_0} + \sum_{t>1} \underbrace{D_{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t))}_{L_{t-1}} + \underbrace{D_{KL}(q(x_T|x_0)||p(x_T))}_{L_T}] \tag{2.17}$$

Para poder tener los fundamentos detrás de la fórmula anterior se recomienda consultar el Apéndice sobre Máxima Verosimilitud.

Para poder entenderla , explicamos a continuación los componentes de la misma:

- NLL es el negativo del logaritmo de la verosimilitud (negative log-likehood).
- Entonces, debemos preguntarnos...¿ qué tan similares son las distribuciones reales q con las que nuestro modelo p_θ logra aprender?. Para ello se utiliza la divergencia KL.
- L_0 es la medición de NLL en el primer paso de la cadena.
- L_{t-1} es la medición de NLL en los pasos intermedios de la cadena.
- L_T se refiere al último paso de la cadena y aquí intervienen dos distribuciones de ruido $\sim \mathcal{N}(0, \mathbf{I})$ que están dadas por el Scheduler, al cuál ya hemos hecho referencia.

Acá la clave es ver que en lugar de entrenar el modelo para predecir la media de la versión sin ruido de x_{t-1} dada la entrada ruidosa x_t, podemos entrenar el modelo para predecir el ruido. Teniendo el ruido podemos calcular la media.(2.15)

Con lo dicho en el párrafo anterior, la pérdida (promediada sobre todo el conjunto de datos) implica en tomar en cuenta los pasos intermedios L_{t-1}. Entonces lo que realmente importa para asegurar una buena aproximación en el proceso de reconstrucción es una estimación precisa de $\epsilon_\theta(x_t, t)$.

Utilizando este argumento, se puede simplicar la Fórmula (2.17) y quedaría de la siguiente forma:

$$\mathcal{L} = \mathbb{E}_{t,x_o,\epsilon}[||\epsilon - \epsilon_\theta(x_t, t)||^2] \tag{2.18}$$

En resumen , es el MSE entre el error real y el error predicho por el modelo (durante todas las etapas intermedias) y por tanto es mucho más manejable pensando en su implementación. Esto es lo que se nombra como L_{simple} en el libro Murphy (2023).

Capítulo 3

Algoritmos

El objetivo de este capítulo es mostrar los principales algoritmos. La finalidad es explicar los mismos y aplicarlos a la conocida base de datos **CelebA** [1].

3.1. Proceso hacia adelante.

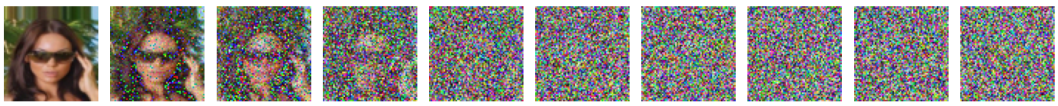
El proceso hacia adelante lo podemos especificar de la siguiente manera:

1. Generamos x_0, muestra inicial de datos reales.
2. Inicializamos T con el números de pasos.
3. Genero el vector β con lista de valores β_t para cada paso t.
4. Genero el vector de α_t a partir de β_t.
5. Inicializo, $x_t = x_0$.
6. Para cada t en $(1..T)$:
   a) Calcula la nueva muestra x_t añadiendo ruido gaussiano a la muestra anterior x_{t-1} usando la siguiente fórmula:
   b) $\sqrt{\bar{\alpha}_t} \cdot x_0 + \sqrt{1 - \bar{\alpha}_t} \cdot \epsilon$,
   c) donde $\bar{\alpha}_t$: es el producto acumulado de α_t hasta el paso t y ϵ es ruido gaussiano aleatorio.

Después de T pasos, la muestra x_T será casi ruido puro. Se muestra a continuación un ejemplo de ejecutar este algoritmo con T=300 usando las imágenes de CelebA.

[1] https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

Figura 3.1: Difusión hacia adelante.



3.2. Proceso hacia atrás.

En esta parte tomamos la última imagen que corresponde a la imagen de la derecha de la figura anterior. A partir de ella la idea es generar la cara de una nueva persona y para ello ya no se utiliza x_0 (imagen inicial).

Para ir hacia atrás en la cadena de imágenes se debe entrenar la red neuronal que permitirá calcular nuestros parámetros de la distribución que buscamos y luego hay que realizar el paso final que consiste en muestrear (sampling) partiendo de imágenes ruidosas o ruido blanco para poder generar una imagen totalmente nueva.

Algorithm 1 Algoritmo de entrenamiento.

1: **repeat**
2: $x_0 \sim q(x_0)$
3: $t \sim Uniform(1, ..., T)$
4: $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Calcular el gradiente de la pérdida con respecto a los parámetros θ : $\nabla_\theta \left\| \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} \cdot x_0 + \sqrt{1 - \bar{\alpha}_t} \cdot \epsilon, t) \right\|^2$
6: Se actualizan los parámetros utilizando un optimizador.
7: **until** se logra convergencia o se llegó a un máximo de iteraciones.

Explicamos a continuación los pasos del algoritmo de entrenamiento:

- Seleccionamos un batch de entrenamiento.
- Generamos aleatoriamente un valor de t distinto para cada muestra.
- Generamos los x_t según el t que le haya tocado a cada uno y guardamos la matriz de ruido de cada una.
- Usamos nuestro modelo para predecir la matriz de ruido en función de x_t y t .

- Calculamos el MSE entre el ruido real y el predicho y retropropagamos el error.
- Repetir hasta obtener la convergencia deseada sobre el error.

Algorithm 2 Algoritmo de Muestreo (Genero nueva imagen).

1: Tomo el modelo entrenado e inicializo $x_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t \leftarrow T$ downto 1 **do**
3: $\quad z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad if \quad t > 1, \quad else \quad z = 0$
4: $\quad x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)) + \Sigma_t z$
5: **end for**
6: **return** x_0, obtengo imagen final.

En este caso ya contamos con la **red neuronal entrenada** por el algoritmo anterior, la cuál está representada por $\epsilon_\theta(x_t, t)$:

- Generar un batch con tantas matrices de ruido aleatorio como muestras queramos generar. Estas serán nuestras x_t.
- Realizar la predicción del ruido que tienen mediante el uso de nuestro modelo utilizando x_t y t.
- Reconstruir x_{t-1} con el ruido predicho.
- Repetir hasta haber recorrido toda la cadena.

Al finalizar este proceso ya tendremos una imagen completamente nueva. Veamos a continuación una imagen obtenida con este proceso hacia atrás.

COLOCAR AQUI IMAGEN.

3.3. xxxxx

La finalidad de una metodología bien descrita es explicitar los pasos mediante los cuales se obtienen los resultados, y por tanto el cumplimiento (o no) de los objetivos establecidos, de manera tal que pueda ser replicado por otro investigador. Si corresponde, también se evaluarán problemas metodológicos y se realizarán consideraciones éticas. En algunas disciplinas este capítulo se denomina Materiales y métodos. En caso de que la investigación sea de carácter experimental, se debe especificar la siguiente información:

- el tipo de investigación realizada (experimental, descriptiva, estudio de caso, encuesta de opinión, etc.)

- el modo de recolección de datos (análisis documental, observación participante o no, entrevista o cuestionario, etc.)
- población o sujetos experimentales.
- protocolo de investigación, si corresponde

En algunas disciplinas formales no es pertinente la inclusión de un capítulo que recoja una cierta metodología de trabajo. En tales casos, se espera que el tesista haga mención de cuestiones carácter metodológico en la Introducción.

El procesamiento del trabajo metodológico que no es imprescindible para la comprensión del texto puede incluirse en apéndices o anexos (Anexo 1).

Capítulo 4

Presentación de los datos, Análisis, Discusión

En algunas disciplinas, el capítulo Presentación de datos va acompañado del análisis o de la discusión de la información (*Presentación y análisis de los datos*; *Resultados y discusión*), en tanto que en otras, *Presentación*, *Análisis* y *Discusión* son capítulos separados. El objetivo de esta(s) parte(s) de la tesis es presentar los datos recabados y el análisis realizado a la luz de la bibliografía ya revisada. Se puede incluir la interpretación de los resultados (*Discusión*) a partir del análisis de los datos, o también relacionarlos con estudios relevantes que se entienden pertinentes, aun si estos no se han consignado en los *Fundamentos teóricos*, ya que se entiende que al analizar los datos pueden aparecer algunos que no se enmarcan teóricamente o que no se explican en el encuadre teórico o en estudios ya existentes.

Ahora a modo de ejemplo mencionamos el símbolo de los números reales utilizando el comando `\gls{}` Real y el comando `\glssymbol{}` $\mathbb{R}$. Otro ejemplo es mencionar el tensor simétrico de tensiones $\boldsymbol{\sigma}$, o un valor escalar α o un conjunto vacío $\emptyset$.

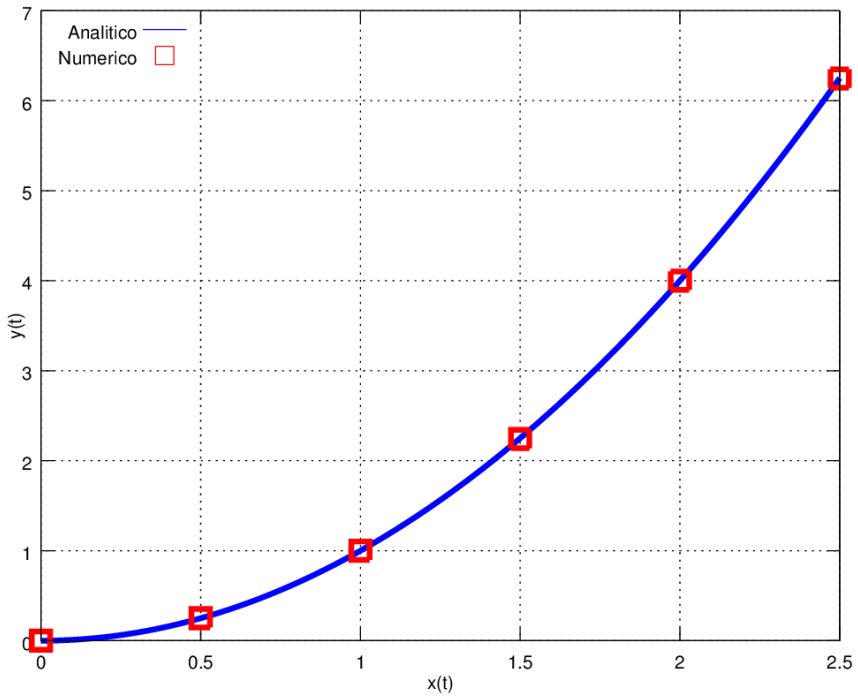
4.1. XXXX

Xxxx
Xxxx

Tabla 4.1: XXXX

t (seg)	x(t)	y(t)
1	0.0000	0.0001
2	0.5000	0.2498
3	1.0000	1.0000
4	1.5000	2.2403
5	2.0000	4.0010
6	2.5000	6.2459

Figura 4.1: XXXX



$$asdasda \tag{4.1}$$

Capítulo 5

Consideraciones finales

En este capítulo se sintetizan las posturas expuestas en el capítulo anterior. Se retoma la pregunta de investigación y se expresa si los resultados apoyan o no la hipótesis planteada.

Además, se pueden hacer contribuciones teóricas o metodológicas a la disciplina y recomendaciones para trabajos futuros o para profundizar en el campo, plantear nuevas interrogantes o proponer explicaciones *post hoc*. En algunos trabajos este capítulo se subdivide en otras secciones que presentan algunos de los contenidos mencionados. En algunas tradiciones académicas este capítulo recibe distintas denominaciones: *Conclusiones*, *Conclusiones y trabajos a futuro*, *Consideraciones finales y recomendaciones*.

Referencias bibliográficas

Jascha Sohl-Dickstein, Eric A. Weiss, N. M. y. S. G. (2015). *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*. Stanford University ; Universidad de California Berkeley, xxxx.

Murphy, K. P. (2022). *Probabilistic Machile Learning, An Introduction*. MIT Press; Cambridge, Massachusetts, London, England.

Murphy, K. P. (2023). *Probabilistic Machile Learning, Advanced Topics*. MIT Press; Cambridge, Massachusetts, London, England.

Tomczak, J. M. (2022). *Deep Generative Modeling*. Springer, Amsterdam.

APÉNDICES

Apéndice 1

Distribución de Gauss o Normal

1.0.1. Introducción

La distribución de Gauss, también llamada distribución Normal, es la más ampliamente usada es estadística y en aprendizaje de máquinas, y esto se debe a varias razones. Veremos algunas a continuación. Para esta parte nos basamos en el libro Murphy (2022).

Primero, ésta tiene dos parámetros que son fáciles de interpretar, los cuáles capturan dos de las propiedades más básicas de la distribución, la media y la varianza. Segundo, el Teorema Central del Límite nos dice que las sumas de las variables aleatorias independientes tiene aproximadamente una distribución Gaussiana, haciendo que ésta sea una buena elección para modelar errores residuales o ruido. Tercero, la distribución Gaussiana hace el menor número de suposiciones (tiene máxima entropía), sujeta a la restricción de ya tener especificadas la media y la varianza. Por último, tiene una forma matemática simple la cuál resulta fácil de implementar.

Para recordar la notación del caso univariado, decimos que X está normalmente distribuida con media μ y desviación estándar σ: $X \sim \mathcal{N}(\mu, \sigma^2)$. En el caso multivariado, de dos o más variables, denotamos la distribución de la siguiente manera: $y \sim \mathcal{N}(\mu, \Sigma)$, dónde $y \in \mathcal{R}^n, \mu \in \mathcal{R}^n$ y Σ es la matriz de covarianza.

1.0.2. Distribución Normal Multivariada

Para expresar matricialmente la probabilidad conjunta ($p(y_1, y_2)$) de dos variables aleatorias gaussianas (y_1) y (y_2), utilizamos la matriz de covarianza (Σ). Supongamos que ($\mathbf{y} = \begin{pmatrix} y_1 & y_2 \end{pmatrix}$) es un vector aleatorio normal bivariado con media ($\mu = \begin{pmatrix} \mu_1 & \mu_2 \end{pmatrix}$) y matriz de covarianza ($\Sigma = \begin{pmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{pmatrix}$).

La función de densidad de probabilidad conjunta para ($\mathbf{y}$) se expresa como:

$p(\mathbf{y}) = \frac{1}{(2\pi)^{k/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{y}-\mu)^T\Sigma^{-1}(\mathbf{y}-\mu)\right)$

Donde:

(k) es la dimensión del vector ($\mathbf{y}$) (en este caso, ($k = 2$)). ($|\Sigma|$) es el determinante de la matriz de covarianza (Σ). (Σ^{-1}) es la inversa de la matriz de covarianza (Σ). ($(\mathbf{y}-\mu)^T$) es la transpuesta del vector ($(\mathbf{y}-\mu)$). En resumen, la probabilidad conjunta de (y_1) y (y_2) se expresa matricialmente utilizando la media y la matriz de covarianza de las variables gaussianas.

1.0.3. Sistemas Gausianos Lineales

En el caso multivariado, sea $z \in \mathcal{R}^L$ un vector desconocido de valores, e $y \in \mathcal{R}^D$ sería algunas medidas ruidosas de de z. Asuminos que dichas variables están relacionadas por la siguiente distribución conjunta:

$$p(z) = \mathcal{N}(z|\mu_z, \Sigma_z)$$
$$p(y|z) = \mathcal{N}(y|\mathcal{W}z + b, \Sigma_y)$$

donde $\mathcal{W}$ es una matriz de dimensión DxL. Este es un ejemplo de sistema linear Gausiano.

El vector conjunto de medias $\mu(z, y)$ lo podemos expresar de la siguiente forma $\begin{pmatrix} \mu_z \\ \mu_y \end{pmatrix} = \begin{pmatrix} \mu_z \\ \mathcal{W}\mu_z + b \end{pmatrix}$.

Calculamos ahora la covarianza de $y = \mathcal{W}z + b$ (dónde W es una matriz de transformación y b es un vector de desplazamiento), aplicando que $Cov(x, y) = \mathbb{E}[(x - \mathbb{E}(x))(y - \mathbb{E}(y))^T]$. Quedando $Cov(y, y) = \Sigma_y = Cov(\mathcal{W}z + b, \mathcal{W}z + b)$, como b es un vector constante, no afecta la covarianza entonces $Cov(y, y) = \Sigma_y = Cov(\mathcal{W}z, \mathcal{W}z)$ y aplicando la propiedad de linealidad de la covarianza,

$Cov(y, y) = \Sigma_y = Cov(\mathcal{W}z, \mathcal{W}z) = \mathcal{W}Cov(z, z)\mathcal{W}^T = \mathcal{W}\Sigma_z\mathcal{W}^T$. La transpuesta ($\mathbf{W}^T$) aparece al final para mantener la simetría y las propiedades de la covarianza.

Introducimos el ruido (proceso hacia adelante)

Pero, hagamos un paréntesis, la variable y se modelará como $Wz + b + \epsilon$ dónde ϵ tiene una distribución $\sim N(0, \Sigma_y)$. Entonces calculemos la $Cov(y, y)$ nuevamente que no puede ser explicada únicamente por la relación lineal con z sino que ahora hay que considerar el **ruido**. Veamos paso a paso, la composición de $Cov(y, y)$:

- La covarianza de y se puede descomponer en dos partes: la covarianza debida a Wz y la covarianza debida al término de ruido ϵ.
- La covarianza de Wz es: $\text{Cov}(Wz) = W\text{Cov}(z)W^T = W\Sigma_z W^T$.
- La covarianza del término de ruido ϵ es simplemente Σ_y, ya que ϵ se asume que tiene una distribución normal con esta covarianza.
- Dado que z y ϵ son independientes, la covarianza total de y es la suma de las covarianzas individuales: $\text{Cov}(y) = \text{Cov}(Wz + \epsilon) = \text{Cov}(Wz) + \text{Cov}(\epsilon) = W\Sigma_z W^T + \Sigma_y$.

Resumiendo, el término Σ_y en la expresión $\text{Cov}(y) = W\Sigma_z W^T + \Sigma_y$ representa la covarianza del término de ruido ϵ que captura la variabilidad en y que no puede ser explicada por la relación lineal con z. Este término es crucial para modelar adecuadamente la incertidumbre en y.

La distribución conjunta como Matrices por Bloques

Sabemos que la forma de expresar $\Sigma(z, y)$ en forma matricial por bloques tiene la siguiente forma

$$\Sigma(z, y) = Cov(z, y) = \begin{pmatrix} \Sigma_z & \Sigma_{zy} \\ \Sigma_{yz} & \Sigma_y \end{pmatrix}$$

, dónde $\Sigma_{zy} = \Sigma_z\mathcal{W}^T$ y $\Sigma_{yz} = \mathcal{W}\Sigma_z$. Veremos de dónde se obtienen las igualdades indicadas a continuación (excepto la de Σ_y que ya explicamos).

$\Sigma_{zy} = Cov(z, y) = \mathbb{E}[(z - \mathbb{E}(z))((\mathcal{W}z + b) - \mathbb{E}([\mathcal{W}z + b])^T)] = \mathbb{E}[(z - \mathbb{E}[z])(\mathcal{W}(z - \mathbb{E}[z])^T)] = \mathbb{E}[(z - \mathbb{E}[z])((z - \mathbb{E}[z])^T)]\mathcal{W}^T = \Sigma_z \mathcal{W}^T$.

$\Sigma_{yz} = Cov(y, z) = \mathbb{E}[(\mathcal{W}z + b - \mathbb{E}[\mathcal{W}z + b])(z - \mathbb{E}[z])^T] = \mathbb{E}[\mathcal{W}(z - \mathbb{E}[z])(z - \mathbb{E}[z])^T] = \mathcal{W}\mathbb{E}[(z - \mathbb{E}[z])(z - \mathbb{E}[z])^T] = \mathcal{W}\Sigma_z$.

Sustituyendo las matrices por bloques con los resultados indicados , tenemos que $\Sigma(z, y) = \begin{pmatrix} \Sigma_z & \Sigma_z \mathcal{W}^T \\ \mathcal{W}\Sigma_z & \Sigma_y + \mathcal{W}\Sigma_z \mathcal{W}^T \end{pmatrix}$.

Entonces la distribución conjunta correspondiente $p(z, y) = p(z)p(y|z)$, es una distribución Gausiana de dimensiones L+D, con media y covarianza dadas por:

$$\mu = \begin{pmatrix} \mu_z \\ \mathcal{W}\mu_z + b + \epsilon \end{pmatrix} \text{ y } \Sigma = \begin{pmatrix} \Sigma_z & \Sigma_z \mathcal{W}^T \\ \mathcal{W}\Sigma_z & \Sigma_y + \mathcal{W}\Sigma_z \mathcal{W}^T \end{pmatrix}$$

Aplicando la fórmula de probabilidad condicional para Gausianas (Murphy (2022) Eq. 3.28 pag 86) a la distribución conjunta $p(y, z)$ podemos computar la posterior $p(z|y)$, como se muestra en la siguiente sección. Esto puede ser interpretado invirtiendo de z por y en el modelo generativo que va en reverso, o sea de variables latentes a observaciones.

1.0.4. Regla de Bayes para Gausianas

Tomando la fórmula 3.28 de Murphy (2022), tenemos que la variable **posterior** z calculada en base a la variable latente y está dada por:

$p(z|y) = \mathcal{N}(z|\mu_{z|y}, \Sigma_{z|y})$
$\Sigma_{z|y}^{-1} = \Sigma_z^{-1} + \mathbf{W}^T \Sigma_y^{-1} \mathbf{W}$
$\mu_{z|y} = \Sigma_{z|y}[\mathbf{W}^T \Sigma_y^{-1}(y - b) + \Sigma_z^{-1}\mu_z]$

Esto es lo que se conoce como **Regla de Bayes para Gausianas** lo cuál se

encuentra detallado con las derivaciones de estas fórmulas en Murphy (2022), sección 3.3.2 .

Distribución condicional: La distribución condicional $p(z|y)$ también será una distribución normal. Para encontrar sus parámetros, usamos las fórmulas de la distribución condicional de una normal multivariada: $p(z|y) \sim N(\mu_{z|y}, \Sigma_{z|y})$ donde: $\mu_{z|y} = \mu + \Sigma_z W^T (W\Sigma_z W^T + \Sigma_y)^{-1}(y - W\mu - b)$ y $\Sigma_{z|y} = \Sigma_z - \Sigma_z W^T (W\Sigma_z W^T + \Sigma_y)^{-1} W\Sigma_z$.
Entonces, La distribución posterior $p(z|y)$ es una normal multivariada con media $\mu_{z|y}$ y covarianza $\Sigma_{z|y}$, dadas por las fórmulas anteriores.

Para calcular $\mu_{z|y}$ y $\Sigma_{z|y}$ en el contexto de un sistema multivariante lineal gaussiano, seguimos las fórmulas derivadas de la distribución condicional de una normal multivariada. Aquí están los pasos detallados:

Parámetros de la Distribución Condicional $p(z|y)$

Media Condicional $\mu_{z|y}$: $\mu_{z|y} = \mu + \Sigma_z W^T (W\Sigma_z W^T + \Sigma_y)^{-1}(y - W\mu - b)$
Covarianza Condicional $\Sigma_{z|y}$: $\Sigma_{z|y} = \Sigma_z - \Sigma_z W^T (W\Sigma_z W^T + \Sigma_y)^{-1} W\Sigma_z$

Apéndice 2

Máxima Verosimilitud

En esta parte veremos cómo aprender los parámetros θ a partir de los datos, $\mathcal{D}$. El proceso de estimar θ de los datos es llamado ajuste del modelo o entrenamiento. Hay varios métodos para producir tales estimaciones, pero la mayoría termina siendo un problema de optimización de la forma:

$\hat{\theta} = \arg \min_{\theta} \mathcal{L}(\theta)$

dónde $\mathcal{L}(\theta)$ es una función de pérdida o función objetivo. Además de calcular $\hat{\theta}$, veremos también el proceso de cómo cuantificar la incertidumbre. Este proceso es denominado inferencia y dentro del campo de aprendizaje profundo se refiere a la predicción que haremos, o sea $p(y|x, \hat{\theta})$. El material de referencia de este apartado se encuentra en Murphy (2022).

2.0.1. Estimación de Máxima Verosimilitud (MLE)

El método más común para estimar parámetros consiste en seleccionar los parámetros que asignan la máxima probabilidad a los datos de entrenamiento, esto es lo que se denomina **estimación de máxima verosimilitud o MLE**.

Definición MLE

Definimos la MLE de la siguiente manera:

$\hat{\theta_{mle}} \triangleq \arg \max_{\theta} p(\mathcal{D}|\theta)$

Asuminos que las muestras de entrenamiento son muestreadas en forma independiente de la misma distribución, por lo tanto la verosimilitud (condicional) sería:

$p(\mathcal{D}|\theta) = \prod_{n=1}^{N} p(y_n|x_n, \theta)$ Se asume que las muestras son **iid**, lo cuál significa **independientes e idénticamente distribuidas.** Se estila trabajar con el **logaritmo de la verosimilitud**, el cuál es dado por la siguiente fórmula:

$$\log(\theta) \triangleq \log p(\mathcal{D}|\theta) = \sum_{n=1}^{N} \log p(y_n|x_n, \theta)$$

El uso del logaritmo descompone el producto en sumas de términos, uno por muestra. Entonces la estimación de máxima verosimilitud, MLE, queda dada de esta forma:

$$\hat{\theta_{mle}} = \arg \max_{\theta} \sum_{n=1}^{N} \log p(y_n|x_n, \theta)$$

Debido a que la mayoría de los algoritmos están diseñados para minimizar las funciones de costo, podemos redefinir la **función objetivo** para que sea el **negativo del logaritmo de la verosimilitud (NLL)**:

$$NLL(\theta)) \triangleq -\log p(\mathcal{D}|\theta) = -\sum_{n=1}^{N} \log p(y_n|x_n, \theta)$$

Minimizando esta expresión se obtendrá el estimador de máxima verosimilitud, MLE. Si el modelo es no supervisado (incondicional), el MLE queda expresado así:

$$\hat{\theta_{mle}} = \arg \min_{\theta} -\sum_{n=1}^{N} \log p(y_n, \theta)$$

debido a que en el caso de no supervisado tenemos los y_n pero no contamos con con las entradas x_n.
Alternativamente podemos querer maximizar la distribución conjunta de entradas y salidas, con lo cuál tenemos :

$$\hat{\theta_{mle}} = \arg \min_{\theta} -\sum_{n=1}^{N} \log p(y_n, x_n|\theta).$$

2.0.2. Justificación para MLE

Nosotros necesitamos que la distribución resultante que vamos a predecir $p(y|\hat{\theta_{mle}})$ sea tan cercana como sea posible (en el sentido que definiremos abajo) a la distribución empírica de los datos. En el caso incondicional la distribución empírica se define por :

$p_{\mathcal{D}}(y) \triangleq \frac{1}{\mathbf{N}} \Sigma_{n=1}^{\mathbf{N}} \delta(y - y_n)$

Observando la fórmula de la distribución empírica se puede ver que es una serie de funciones delta o picos en los puntos de entrenamiento observados. Se quiere crear un modelo cuya distribución $q(y) = p(y|\theta)$ sea similar a $p_{\mathcal{D}}(y)$.
Una forma estándar de medir la distancia o entre distribuciones de probabilidad p y q es la **divergencia Kullback Leibler (KL)**. La misma se puede definir como
$\mathbf{D}_{\mathcal{KL}}(p||q) = \sum_y p(y) log \frac{p(y)}{q(y)} = \sum_y p(y) \log p(y) - \sum_y p(y) \log q(y)$.

En esta última igualdad tenemos dos sumatorias que se restan, la primera de ellas es la **entropía** negativa de p ($\mathbb{H}(p)$) y la segunda es la entropía cruzada de p y q ($\mathbb{H}_{ce}(p, q)$).

Podemos ver que, la entropía cruzada puede expresarse como la suma de la entropía de p y la divergencia KL entre p y q
$\mathbb{H}(p, q) = \mathbb{H}(p) + \mathbf{D}_{KL}(p||q)$.

Esto significa que la entropía cruzada incluye tanto la incertidumbre inherente en p como la discrepancia entre p y q. Es muy útil en ML, como función de pérdida u objetivo, dónde se busca minimizar la entropía cruzada para mejorar la precisión del modelo.

Volviendo a nuestras fórmulas, si definimos $q(y) = p(y|\theta)$ e igualamos $p(y) = p_{\mathcal{D}}(y)$, la divergencia KL se puede expresar así

$$\mathbf{D}_{\mathcal{KL}}(p||q) = \Sigma_y [p_{\mathcal{D}}(y) \log p_{\mathcal{D}}(y) - p_{\mathcal{D}}(y) \log q(y)]$$
$$= -\mathbb{H}(p_{\mathcal{D}}) - \frac{1}{\mathbf{N}} \Sigma_{n=1}^{\mathbf{N}} \log p(y_n|\theta) = const + NLL(\theta)$$

El primer término es una constante que podemos ignorar, entonces nos queda para considerar el término NLL solamente. Entonces, minimizando la divergencia KL es equivalente a minimizar NLL (el negativo de la máxima verosimilitud) lo cuál es equivalente a calcular el MLE ,
$\hat{\theta_{mle}} = \arg\min_\theta - \sum_{n=1}^{N} \log p(y_n, x_n|\theta)$.

2.0.3. Ejemplo: MLE para la Gausiana Multivariada

Ya se ha expuesto que hay un proceso de hallar el máximo o el mínimo de una función dada para calcular la MLE.

La optimización es un proceso matemático que busca encontrar el mejor valor (máximo o mínimo) de una función objetivo, dadas ciertas restricciones.

La estimación de máxima verosimilitud es un método estadístico para estimar los parámetros de un modelo probabilístico, maximizando la función de verosimilitud.

Por lo tanto parece conveniente recordar en la siguiente subsección cómo realizar ese proceso de optimización.

Recordamos un poco de Optimización.

Para poder entender porqué hacemos ciertos cálculos hay que recordar para qué sirven las derivadas y qué son los puntos críticos.

- Derivada: La derivada de una función en un punto mide la tasa de cambio de la función en ese punto. Si la derivada es positiva, la función está aumentando; si es negativa, la función está disminuyendo.
- Puntos Críticos: Los puntos críticos son aquellos donde el gradiente ∇f de la función es cero. Matemáticamente, si ($\nabla f(x) = 0$), entonces (x) es un punto crítico.
- Cuando el gradiente es cero en un punto crítico, significa que la función tiene un plano tangente horizontal en ese punto.

Si el gradiente es cero, esto quiere decir :

- Máximo Local: Un punto donde la función alcanza un valor máximo en una vecindad.

- Mínimo Local: Un punto donde la función alcanza un valor mínimo en una vecindad.
- Punto Silla: Existen direcciones en las cuáles la función crece, y en otras direcciones, decrece.

Aplicando lo expuesto en el contexto de la estimación de máxima verosimilitud (MLE), encontrar los puntos críticos de la función de verosimilitud (donde el gradiente es cero) nos ayuda a identificar los valores de los parámetros que maximizan la verosimilitud de los datos. Esto es crucial para ajustar el modelo de manera óptima a los datos.

MLE para la media

Derivar la función de verosimilitud respecto a μ es un paso crucial en la estimación de máxima verosimilitud (MLE) porque nos permite encontrar el valor de μ que maximiza la probabilidad de observar los datos dados. Veremos a continuación el razonamiento para obtener la MLE para la media.

La idea central de MLE es encontrar los parámetros del modelo (en este caso, μ y Σ que hacen que los datos observados sean lo más probables posible. Para una distribución gaussiana multivariada, esto implica maximizar la función de verosimilitud $L(\mu, \Sigma)$.

Para maximizar una función, buscamos sus puntos críticos, que son los puntos donde la derivada de la función es cero. En el contexto de MLE, derivamos la log-verosimilitud respecto a μ para encontrar estos puntos críticos. La derivada de la log-verosimilitud respecto a μ nos da una ecuación que podemos resolver para μ.

Proceso de Derivación

- Log-Verosimilitud: Tomamos el logaritmo de la función de verosimilitud para simplificar los cálculos.
- Derivada: Derivamos la log-verosimilitud respecto a μ.
- Igualar a Cero: Igualamos la derivada a cero para encontrar los puntos críticos.
- Resolver para μ: Resolviendo la ecuación obtenida, encontramos el valor de μ que maximiza la verosimilitud.

Se detalla a continuación el cálculo de la MLE para la media:

- Paso 1) Definir la Función de Verosimilitud : dada una muestra de datos $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n$, donde cada $\mathbf{y}_i$ es un vector de dimensión (d), la función de densidad de probabilidad de una distribución gaussiana multivariada es:
  $p(\mathbf{y}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{y}-\mu)^T\Sigma^{-1}(\mathbf{y}-\mu)\right)$

  La función de verosimilitud para el conjunto de datos es el producto de las densidades individuales: $L(\mu, \Sigma) = \prod_{i=1}^{n} p(\mathbf{y}_i)$

- Paso 2) Logaritmo de la Función de Verosimilitud. Para simplificar el cálculo, tomamos el logaritmo de la función de verosimilitud:
  $\log L(\mu, \Sigma) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log|\Sigma| - \frac{1}{2}\sum_{i=1}^{n}(\mathbf{y}_i-\mu)^T\Sigma^{-1}(\mathbf{y}_i-\mu)$
- Paso 3) Derivar Respecto a μ : Derivamos la log-verosimilitud respecto a μ y la igualamos a cero para encontrar el punto crítico:
  $\frac{\partial}{\partial\mu}\log L(\mu, \Sigma|X) = \Sigma^{-1}\sum_{i=1}^{n}(x_i-\mu) = 0$

Y resolviendo para μ:

- Partimos de la ecuación dada $\Sigma^{-1}\sum_{i=1}^{n}(x_i-\mu) = 0$ y multiplicamos ambos lados por Σ para eliminar Σ^{-1} : $\sum_{i=1}^{n}(x_i-\mu) = 0$.
- Distribuimos la suma $\sum_{i=1}^{n} x_i - \sum_{i=1}^{n}\mu = 0$, dado que μ es constante, podemos sacarlo de la suma: $\sum_{i=1}^{n} x_i - n\mu = 0$.
- Despejamos μ : $n\mu = \sum_{i=1}^{n} x_i$,entonces $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$.

Por lo tanto, la solución para μ es el promedio de las observaciones x_i : $\mu = \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, dicho de otra forma es la media empírica de dichas observaciones.

MLE para la matriz de Covarianza

Para deducir la matriz de covarianza que maximiza la verosimilitud en una distribución normal multivariada, seguimos estos pasos:
Función de verosimilitud: La función de verosimilitud para una muestra $X = x_1, x_2, \ldots, x_n$ de una distribución normal multivariada $N(\mu, \Sigma)$ es: $L(\mu, \Sigma|X) = \prod_{i=1}^{n} \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x_i-\mu)^T\Sigma^{-1}(x_i-\mu)\right)$ donde d es la dimensión de los vectores x_i.

Log-verosimilitud: Tomamos el logaritmo de la función de verosimilitud para

simplificar los cálculos: $\log L(\mu, \Sigma|X) = -\frac{nd}{2}\log(2\pi) - \frac{n}{2}\log|\Sigma| - \frac{1}{2}\sum_{i=1}^{n}(x_i - \mu)^T\Sigma^{-1}(x_i - \mu)$.

Derivada respecto a Σ: Calculamos la derivada de la log-verosimilitud con respecto a Σ: $\frac{\partial}{\partial\Sigma}\log L(\mu, \Sigma|X) = -\frac{n}{2}\Sigma^{-1} + \frac{1}{2}\Sigma^{-1}\left(\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T\right)\Sigma^{-1}$

Igualamos a cero: Igualamos la derivada a cero para encontrar el punto crítico: $-\frac{n}{2}\Sigma^{-1} + \frac{1}{2}\Sigma^{-1}\left(\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T\right)\Sigma^{-1} = 0$.

Simplificación: Multiplicamos ambos lados por Σ y simplificamos: $n\Sigma = \sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T$.

Solución para Σ: Despejando Σ: $\Sigma = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T$. Por lo tanto, la matriz de covarianza que maximiza la verosimilitud es la matriz de covarianza muestral: $\Sigma = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T$.

Esta matriz de covarianza muestral es un estimador insesgado de la verdadera matriz de covarianza de la distribución normal multivariada ; porque, en promedio, su valor esperado es igual al valor verdadero de la matriz de covarianza de la población.

========== Hasta aquí ========== Definición de Estimador Insesgado Un estimador $\hat{\theta}$ de un parámetro θ es insesgado si su valor esperado es igual al parámetro que está estimando: $E[\hat{\theta}] = \theta$.

$E[\hat{\Sigma}] = E\left[\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T\right]$

Dado que x_i son muestras independientes de una distribución normal multivariada con media μ y matriz de covarianza Σ, podemos descomponer la expresión: $E[\hat{\Sigma}] = \frac{1}{n}\sum_{i=1}^{n}E[(x_i - \bar{x})(x_i - \bar{x})^T]$.

Usando propiedades de la esperanza y la varianza, y considerando que $\bar{x}$ es un estimador insesgado de μ, se puede demostrar que: $E[(x_i - \bar{x})(x_i - \bar{x})^T] = \Sigma$.

Por lo tanto: $E[\hat{\Sigma}] = \Sigma$

Dado que el valor esperado de la matriz de covarianza muestral $\hat{\Sigma}$ es igual a la verdadera matriz de covarianza Σ, podemos concluir que $\hat{\Sigma}$ es un estimador insesgado de Σ.

Apéndice 3

Modelos de Variables Latentes

Se toma como referencia el siguiente libro Tomczak (2022), el cuál expone un compendio de las técnicas usadas actualmente para los modelos generativos.
Dentro del marco de la generación de imágenes, supongamos que contamos con un conjuntos de imágenes de caballos por ejemplo. Nuestro interés es aprender $p(x)$, para poder generar nuevas imágenes de caballos.

Usaremos las matemáticas para expresar el proceso de generar imágenes. Comenzamos con nuestras objetos de interés de alta dimensión, $x \in \mathcal{X}^D$ (por ej., imágenes $\mathcal{X} \in 0, 1, ..., 255$) y variables latentes, $z \in \mathcal{Z}^M$ (por ej., $\mathcal{Z} = \mathcal{R}$), las que podemos llamamos factores ocultos en los datos.

Entonces, el proceso generativo puede ser expresado de la siguiente manera:

- $z \sim p(z)$
- $x \sim p(x|z)$

Traduciendo lo anterior, primero tomamos una muestra de z y entonces creamos una nueva imagen, dicho de otra forma, tomamos una muestra x de la distribución condicional $p(x|z)$. Este último proceso es el proceso generativo o generador.

El modelo más ampliamente conocido de variables latentes es el **Análisis de Componentes Principales Probabilístico** (pPCA) dónde $p(z)$ y $p(x|z)$ tiene una distribución Gaussiana, y la dependencia entre z y x es lineal. La opción no lineal de pPCA con distribuciones arbitrarias es la denominada **Variatio-**

nal Auto-Encoder (VAE). Para hecer manejable el proceso de inferencia, se utiliza la inferencia variacional para aproximar la estimada (posterior) $p(z|x)$ y redes Neuronales son las usadas para parametrizar la distribución.

ANEXOS

Anexo 1

Material legislativo

xxxxx