USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA
FACULTADE DE MATEMÁTICAS

Traballo Fin de Grao

Modelos gráficos e redes neuronais para a estimación da densidade en alta dimensión

Manuel de Prada Corral

Xullo, 2022

UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

GRAO DE MATEMÁTICAS

Traballo Fin de Grao

Modelos gráficos e redes neuronais para a estimación da densidade en alta dimensión

Manuel de Prada Corral

Xullo, 2022

UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Traballo proposto

Área de Coñecemento: Estatística e Investigación Operativa
Título: Modelos gráficos e redes neuronais para a estimación da densidade en alta dimensión
Breve descrición do contido
Este traballo ten como marco a análise estatística de distribucións complexas con alta dimensionalidade, habitualmente tratadas con métodos de aprendizaxe automática (machine learning). Para realizar tarefas de mostraxe e de estimación de densidade, na aproximación clásica de modelos paramétricos, impóñense condicións que faciliten os cálculos, xeralmente realizados mediante técnicas de máxima verosimilitude. O obxectivo deste TFG é afondar nos conceptos e metodoloxías máis utilizados para a estimación da densidade nun contexto de alta dimensionalidade. En concreto, ademais doutras posibles aproximacións, revisaranse as ideas clásicas da estimación por máxima verosimilitude de modelos paramétricos (así como as limitacións da estimación tipo núcleo neste campo e a posibilidade de empregar metodoloxías semiparamétricas ou de mesturas de distribucións), e as distintas estratexias que se poden considerar para mellorar e simplificar o proceso de estimación. De xeito específico estudaranse os modelos gráficos, e as súas vantaxes e limitacións a través de NADE (Neural Autoregressive Density Estimation) e outros exemplos. Os conceptos introducidos ilustraranse mediante simulación e exemplos prácticos. Compararanse os diferentes métodos de estimación xerando mostras a partir das distribución extraídas de conxuntos de datos de imaxes ou tabulares.

Índice

Resumo

Os modelos gráficos son modelos estatísticos que se apoian na Teoría de Grafos para modelar interaccións complexas entre variables aleatorias. Neste traballo estúdanse as súas bases teóricas e a aplicación dos mesmos na estimación de densidade en contextos de alta dimensión. Comezamos introducindo as técnicas clásicas de estimación de densidade non paramétrica e ilustrando as dificultades que aparecen empregando un conxunto de datos real. A continuación, desenvólvese a teoría dos modelos gráficos, tomando como exemplo de referencia a máquina de Boltzmann restrinxida.

A partir deste caso, preséntase NADE, un estimador que emprega redes neuronais para superar varias das dificultades que xorden coa máquina de Boltzmann restrinxida. Finalmente, compáranse experimentalmente mediante simulacións os modelos presentados ao longo do traballo e discútense os resultados e as liñas de investigación abertas.

Abstract

Graphical models are statistical models that rely on Graph Theory to model complex interactions between random variables. This project studies the theoretical basis of these models and their application to density estimation in high dimensional contexts. We first introduce the classical non-parametric density estimation techniques and illustrate the challenges that arise when using a real dataset. Then, the graphical models' theoretical background is developed, taking the restricted Boltzmann machine as a reference example.

Following this example, we present NADE, a distribution estimator that leverages neural networks to overcome several difficulties present in the restricted Boltzmann machine. Finally, we compare experimentally through simulation the presented models and discuss the results and open research directions.

Introdución

Na actualidade, a Estatística aborda o estudo de datos e información cada vez máis complexos no contexto da transformación dixital que estamos a vivir. O exponencial crecemento nas capacidades computacionais posibilita a análise de imaxes, series biomédicas ou texto, aproveitando as ferramentas da estatística clásica. O potencial das conclusións extraídas ao modelar unha realidade tan complexa é enorme. Non obstante, este contexto de datos masivos trae consigo restricións especiais atendendo sobre todo á formalización dos modelos e á selección de ferramentas.

Neste traballo trataranse varias vías de aproximación á realización de inferencia estatística sobre datos masivos de alta dimensión. Ademais da discusión teórica, experimentarase con modelos concretos e conxuntos de datos de referencia. No Capítulo 1 estudaranse as ferramentas básicas de estimación como os métodos núcleo ou as mesturas de distribucións. Ademais, discutiranse os problemas que xorden de intentar aplicar estes métodos no contexto deste traballo, e introduciranse outros conceptos teóricos fundamentais para os desenvolvementos posteriores.

A continuación, no Capítulo 2 realízase un percorrido polos fundamentos teóricos e prácticos dos modelos gráficos. Introdúcense os diferentes tipos, as problemáticas que presenta facer inferencia e simulación e as distintas solucións propostas na literatura. Para rematar esta introdución teórica, fixámonos nas redes autorregresivas loxísticas, que serven de introdución para o caso de estudo deste traballo, Neural Autoregressive Distribution Estimation (NADE).

O Capítulo 3 desenvolve NADE, un modelo gráfico moi exitoso baseado nos principios de varios modelos gráficos presentados no Capítulo anterior. Estudarase a motivación da súa parametrización, que emprega redes neuronais, e analizarase comparativamente os resultados que obtén estimando a densidade de datos de alta dimensión.

Finalmente, o Capítulo 4 discutirá os principais logros e problemas de NADE, tanto na aproximación xeral ao problema como nos detalles de implementación e decisións de deseño. Tamén se analizarán as posibles liñas de investigación abertas para o futuro.

Na experimentación e discusión de resultados empregarase o conxunto de datos de referencia

MNIST recopilado en LeCun et al. (2010), que recolle 70.000 imaxes de díxitos numéricos escritos a man, sen cor e cunha resolución de 28×28 píxeles. As imaxes represéntanse numericamente como unha matriz de 28x28 que toma valores entre 0 e 1, en función da intensidade do píxel. A Figura 1 amosa exemplos de dito conxunto de datos.

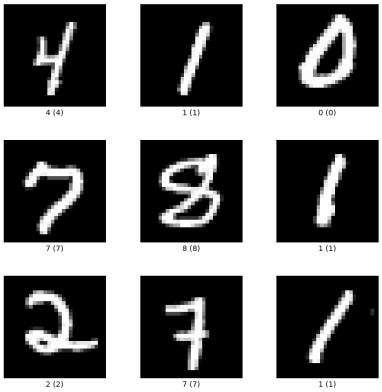


Figura 1: Mostras do conxunto MNIST. As cifras foron obtidas polo NIST de documentos da Oficina do Censo dos EE. UU.

Nos Capítulos 2 e 3 simularanse díxitos novos do conxunto MNIST empregando os modelos estudados. Para iso, adaptaranse as implementacións existentes e actualizarase o código segundo os estándares modernos de redes neuronais, de xeito que o adestramento se leve a cabo en tarxetas gráficas. Con tal fin, empregarase *Theano*, un paquete de software para *Python* presentado en Al-Rfou et al. (2016).

A referencia Goodfellow et al. (2016) aborda de xeito introdutorio a maioría de conceptos tratados neste traballo. A maiores, Koller e Friedman (2009) proporciona un estudo exhaustivo de todas as cuestións relacionadas con modelos gráficos. Finalmente, Uria et al. (2016) desenvolve o caso de estudo do modelo NADE (Capítulo 3). Esta contribución presenta unha revisión da publicación orixinal, que data do 2011, e inclúe os traballos posteriores que estenderon a NADE comentados polos autores.

Capítulo 1

Conceptos previos

Neste capítulo introdúcense as ferramentas clásicas de estimación de densidade nun contexto non paramétrico. Na primeira sección, presentamos a función de densidade, xunto con dous procedementos de estimación alternativos aos modelos paramétricos: a estimación tipo núcleo e as mesturas de distribucións. Estes contidos desenvólvense na materia *Inferencia Estatística* do Grao en Matemáticas. Tras esta primeira aproximación á estimación da densidade en baixa dimensión, abórdase a extensión a alta dimensión, formulando os problemas que presentan os métodos clásicos. A continuación, defínense a función de verosimilitude, a diverxencia de Kullback–Leibler e a entropía cruzada. Todos estes son conceptos fundamentais para a construción de ferramentas de estimación. Finalmente, introdúcense as redes neuronais e achégase un exemplo do seu emprego como ferramenta de estimación de densidade.

1.1. A función de densidade. Ferramentas para a súa estimación

De xeito intuitivo, unha variable aleatoria X é unha función que fai corresponder os posibles resultados dun experimento aleatorio con valores nun espazo medible, o soporte. Exemplos clásicos poderían ser asignar un número ás caras dun dado (variable aleatoria que toma valores discretos) ou as estaturas en centímetros dunha poboación animal (con valores continuos). Dada unha variable aleatoria, as distribucións de probabilidade pretenden asignar un reparto teórico da probabilidade que ten de tomar nas distintas rexións do soporte. No exemplo das estaturas en centímetros, a variable aleatoria tomará un valor en $[0, \infty) \subset \mathbb{R}$, e unha distribución informará sobre a probabilidade que ten unha observación de caer nos distintos subconxuntos do soporte.

Estas nocións intuitivas formalízanse nas funcións de distribución e densidade dunha variable aleatoria X. A función de distribución defínese como a probabilidade de que X tome un valor menor ou igual ao dado, é dicir, $F_X(x) = \mathbb{P}(X \leqslant x)$, para $x \in \mathbb{R}$. Traballaremos con variables

aleatorias absolutamente continuas, que garanten a existencia dunha función de densidade. Neste caso, a información da distribución de probabilidade exprésase identicamente utilizando a función de densidade, f, que se define como:

$$f(x) = F'(x) \text{ e } F(x) = \mathbb{P}(X \leqslant x) = \int_{-\infty}^{x} f(u)du, \text{ para case todo } x \in \text{sop}(X),$$

onde $\text{sop}(X)$ denota o soporte de X e a expresión "para case todo" refírese a que se verifica para todos os puntos salvo un subconxunto de medida de Lebesgue nula. Xeometricamente, nun soporte continuo a probabilidade individual de cada punto é nula, pero a función de densidade expresa como de probable é que unha mostra caia *preto* dun punto dado. No caso multidimensional, a función de distribución e a de densidade conxunta dun vector aleatorio $\boldsymbol{X} = (X_1, \dots X_n)$ defínense, por extensión do caso unidimensional, como

$$F(x_1, \dots, x_n) = \mathbb{P}(X_1 \leqslant x_1, \dots, X_n \leqslant x_n) = \int_{(-\infty, x_1)\times\cdots\times(-\infty, x_n)} f(u_1, \dots, u_n)du_1 \dots du_n.$$

Ilustración sobre o conxunto de datos. Traslademos agora estes conceptos, formulados inicialmente nunha variable aleatoria con soporte real, aos conxuntos de datos que presentamos na introdución. No caso do MNIST, o soporte son todos os posibles valores dunha matriz de 28×28 con dominio $[0, 1] \subset \mathbb{R}$, ou, de xeito equivalente, $[0, 1]^{784} \subset \mathbb{R}^{784}$. Se puidesemos debuxar a función de densidade, veríamos como nunha parte importante do soporte, a probabilidade sería moi próxima a 0, xa que a maioría de combinacións aleatorias en $[0, 1]^{784}$ non xeran díxitos manuscritos verosímiles. Observaríamos tamén como, en rexións moi illadas deste espazo, concéntrase toda a probabilidade. En particular, nas 10 contornas do soporte correspondentes aos díxitos numéricos teríamos un pico na función de densidade. Apreciaríamos tamén como, aínda que estas rexións con máis probabilidade están en xeral bastante illadas, habería *camiños de probabilidade* apreciables entre certos díxitos que están moi próximos na súa forma manuscrita, por exemplo o 4 e 9.

Sigamos comparando os exemplos e supoñamos que dispoñemos dunha estimación da función de densidade, $f : [0, 1]^{784} \longrightarrow \mathbb{R}^+$. Integrar esta función numericamente para obter a función de distribución ou outras medidas acumulativas útiles sería moi complicado pola alta dimensionalidade do soporte, como veremos en capítulos posteriores. Incluso estimar a propia función de densidade é intratable empregando métodos tradicionais, como discutiremos a continuación.

Pese ás dificultades, segue sendo moi proveitoso realizar inferencia sobre estes conxuntos de datos. Queremos poder simular novas observacións destes conxuntos, ou identificar como de probable é que certa imaxe se corresponda coa distribución. Así mesmo, coñecer a distribución revela información sobre a estrutura intrínseca dos datos que facilita outras tarefas como a clasificación do díxito observado, no caso do MNIST.

Esta clase de conxuntos de datos, en xeral, non seguen ningún tipo de distribución de probabilidade paramétrica xa coñecida. Serán necesarios os métodos non paramétricos de estimación de densidade, que non asumen ningunha estrutura previa e traballan unicamente coa información presente nos datos. A continuación, preséntanse os fundamentos destes métodos.

1.1.1. Estimación tipo núcleo

A primeira aproximación á estimación de densidade non paramétrica é o histograma, que simplemente representa as frecuencias relativas dos datos dividindo o soporte en intervalos. Ten varios problemas: é descontinuo, depende fortemente do parámetro de fiestra h que determina a agrupación dos datos en intervalos e depende tamén do punto inicial escollido para construír os intervalos. A súa expresión como estimador da función de densidade é:

$$H(x) = \frac{1}{nh} \sum_{i=1}^{n} I_{\{X_i \in I_m\}}, \text{ se } x \in I_m,$$

sendo $I_m = (x_0 + hm, x_0 + h(m+1)]$ e $m \in \{0, \dots, M-1\}$ tal que $x_0 < X_{(1)} \leq \cdots \leq X_{(n)} \leq x_0 + hM$, onde $X_{(\cdot)}$ denota á mostra ordenada. A evolución natural deste estimador é o histograma móbil ou estimador naive, que se define como

$$H_m(x) = \frac{1}{2nh} \sum_{i=1}^{n} I_{\{X_i \in (x-h, x+h)\}}, \quad \forall x \in \mathbb{R}.$$

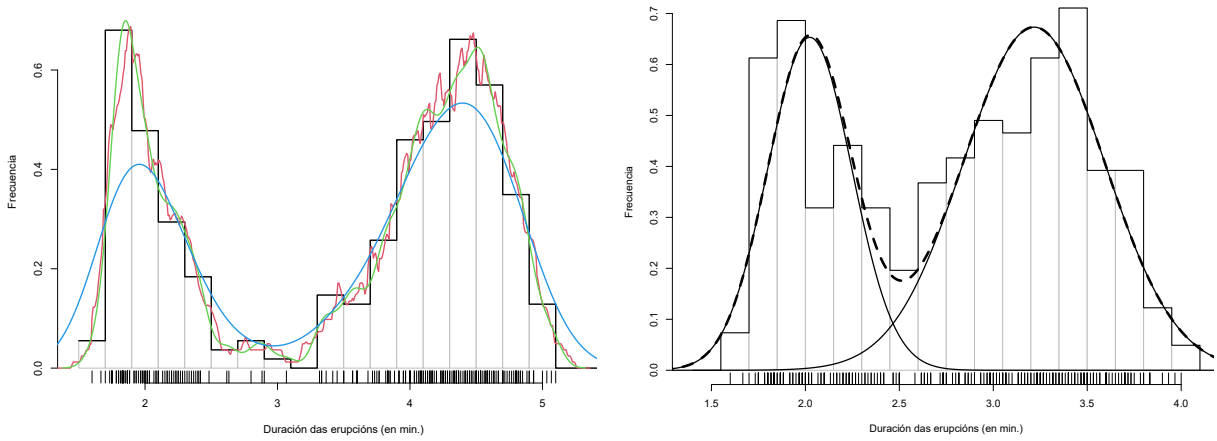
Esta aproximación xa non necesita dun punto inicial x_0. Segue sen ser continua, pois a conta que realiza o indicador I é fundamentalmente discreta, pero xa proporciona unha estimación da densidade moi razoable.

Os estimadores tipo núcleo nacen do seguinte paso natural: substituír a función indicadora por unha alternativa continua. Defínense como:

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} \mathcal{K}\left(\frac{x - X_i}{h}\right), \quad \forall x \in \mathbb{R}.$$

A función núcleo $\mathcal{K}$ determina as propiedades do estimador. Polo tanto, debe ser unha función de densidade e, dependendo do contexto, soe ser desexable que sexa regular e simétrica respecto ao cero. Unha elección habitual é a función de densidade da normal estándar, denominada núcleo gaussiano. Na Figura 1.1a pódese observar a diferencia entre os diferentes métodos de estimación expostos e o efecto do parámetro de fiestra ou de suavizado h sobre a estimación tipo núcleo.

Para un desenvolvemento en profundidade da estimación tipo núcleo véxase Wand e Jones (1994).



(a) (b)

Figura 1.1: Na esquerda, ilústranse diferentes representacións da densidade para os datos *Old Faithful* incluídos no software estatístico R. En negro, un histograma con $h = 0{,}3$. En vermello, un histograma móbil con $h = 0{,}3$. En verde e azul, dous núcleos gaussianos con $h = 0{,}3$ e $h = 1$, respectivamente. A figura da dereita amosa a mestura de dúas gaussianas, empregando unha perturbación dos datos anteriores a efectos ilustrativos.

1.1.2. Mesturas de distribucións e variables latentes

As mesturas de distribucións constrúense combinando varias distribucións máis simples e son especialmente proveitosas para estimar en contextos asimétricos e/ou multimodais, como o exemplo da Figura 1.1b. Nunha mestura, considéranse K compoñentes (habitualmente, serán normais $N(\mu_k, \sigma_k)$, con $k = 1, \dots, K$) e unha distribución discreta categórica. Esta distribución discreta, tamén chamada multinoulli por ser unha xeneralización da Bernoulli, describe unha variable aleatoria que pode tomar como valor unha de K categorías posibles, con probabilidades $(\pi_1, \pi_2, \dots, \pi_K)$ para cada categoría. Nunha mestura de distribucións, considérase que unha mostra da multinoulli determina cal das K compoñentes da mestura será a encargada de producir a mostra. Polo tanto, cada dato da mostra da mestura ten asociada unha variable descoñecida á que chamaremos δ, que determina a compoñente da mestura da que provén.

Este exemplo serve para ilustrar o concepto de *variable latente*. Dada unha distribución da que temos unha mostra, unha variable latente ou oculta é unha variable que non podemos observar directamente no resultado do experimento, pero que ten influencia na mostraxe. No exemplo da Figura 1.1b, se facemos unha observación con valor 4, sabemos que case con total seguridade, a variable latente "compoñente da mestura" asociada á observación será a segunda. Sen embargo, se a observación resulta ser un 2.5, non podemos inferir con alta probabilidade o valor da variable latente, pois a observación podería vir das dúas compoñentes con probabilidade similar.

Formalicemos agora as mesturas de distribucións. Dada unha mostra $\boldsymbol{X} = (X_1, \dots, X_n)$ de n observacións independentes, cada observación X_i foi extraída dunha das K distribucións compoñentes da mestura. Os parámetros da multinoulli $(\pi_1, \pi_2, \dots, \pi_K)$, denomínanse "proporcións da mestura" e codifican a probabilidade de que unha observación X_i proveña da compoñente k-ésima da mestura. Para simular unha observación, haberá que obter da multinoulli unha compoñente $\delta = k \in \{1, \dots, K\}$ e posteriormente simular unha observación na compoñente k-ésima. A función de densidade da mestura será:

$$f(x) = \sum_{k=1}^{K} \pi_k f_k(x), \text{ con } \sum_{k=1}^{K} \pi_k = 1.$$

As mesturas de distribucións son unha gran ferramenta para estimar a densidade de nubes de puntos multimodais, que se agrupan en conxuntos máis ou menos illados. Con este fin, soe considerarse unha mestura de distribucións normais, denominada mestura gaussiana:

$$f(x) = \sum_{k=1}^{K} \pi_k N(x; \mu_k, \sigma_k^2) = \sum_{k=1}^{K} \pi_k \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left\{-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right\}. \tag{1.1}$$

Trátase dun método de estimación semiparamétrico, xa que o valor de K condiciona o número de parámetros e a selección de K é complexa. Unha vez fixado K, estimar a densidade con esta aproximación redúcese a optimizar para $\theta = \{\mu_1, \dots, \mu_K, \sigma_1, \dots, \sigma_K, \pi_1, \dots, \pi_K\}$. Na práctica, estes parámetros pódense estimar mediante métodos baseados en técnicas de máxima verosimilitude, que serán introducidas na próxima sección.

Finalmente, recordemos a expresión dun estimador núcleo con núcleo gaussiano estándar:

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{(\frac{x-X_i}{h})^2}{2}\right\} = \sum_{i=1}^{n} \frac{1}{n} \frac{1}{\sqrt{2\pi h^2}} \exp\left\{-\frac{(x-X_i)^2}{2h^2}\right\}.$$

Comparando coa ecuación (1.1), observamos como, para unha mostra aleatoria simple de n observacións, un estimador núcleo baseado na densidade dunha normal estándar non é máis que unha mestura de $K = n$ distribucións normais con $\mu_k = X_i$, $\sigma_k = h$ e $\pi_k = \frac{1}{n} = \frac{1}{K}$.

As mesturas de normais empregaranse no Capítulo 3, xa que proporcionan unha maneira moi sinxela de estender un modelo multidimensional sobre variables discretas ao caso continuo. Como veremos máis adiante, NADE é un modelo de variables aleatorias binarias que simula mostras producindo para cada variable o parámetro dunha Bernouilli (probabilidade de que esa variable adopte o valor 1). Como veremos, a extensión a valores continuos producirá para cada variable as medias e varianzas dunha mestura de distribucións normais.

1.1.3. A maldición da dimensión

O termo "maldición da dimensión" (do inglés "curse of dimensionality") refírese ao conxunto de problemas e fenómenos que xorden ao traballar con centos ou miles de dimensións, en di-

ferentes campos como a computación, a análise numérica, a estatística ou a minería de datos. Moitos destes problemas pódense entender intuitivamente observando que o "volume" dun espazo aumenta exponencialmente con cada nova dimensión. En consecuencia, requírense cada vez máis datos para manter a densidade dos mesmos.

No caso da estimación de densidade, a necesidade de máis datos é unha das implicacións máis inmediatas. Ao aumentar a dimensión, cada vez precísase dunha mostra máis numerosa para manter a mesma marxe de erro sobre a función de densidade estimada. Por exemplo, no Capítulo 7 de Scott (2015) estúdase mediante simulación o caso do estimador núcleo gaussiano, que precisa de máis de $n = 10^6$ observacións en 10 dimensións para manter unha marxe de erro teórico similar á que conseguía con $n = 50$ observacións nunha dimensión. Supoñendo que temos a cantidade de datos necesaria, este fenómeno implica un incremento exponencial na capacidade de cómputo necesaria para obter unha estimación efectiva da densidade.

O número de mostras non é o único problema que aparece na estimación tipo núcleo ao escalar a dimensión. A expresión do estimador tipo núcleo con dimensión d é

$$\hat{f}(\boldsymbol{x}) = \frac{1}{n|\boldsymbol{H}|} \sum_{i=1}^{n} \mathcal{K}(\boldsymbol{H}^{-1}(\boldsymbol{x} - \boldsymbol{x}_i)),$$

onde $x \in \mathbb{R}^d$, $\mathcal{K} : \mathbb{R}^d \longrightarrow \mathbb{R}$ é unha función de densidade multidimensional e $\boldsymbol{H}$ é unha matriz $d \times d$ simétrica e non singular. A matriz $\boldsymbol{H}$ é o análogo ao parámetro de fiestra h, pero agora son $\frac{d(d-1)}{2}$ valores a optimizar. No exemplo do parágrafo anterior, en Scott (2015) considérase para a simulación $\boldsymbol{H} = h \cdot \boldsymbol{I}_d$, con $h \in \mathbb{R}$, pero isto é unha simplificación pouco xeral que non se axusta á realidade. Pensemos no conxunto de datos MNIST, cuxos datos teñen dimensión 784 procedentes dunha matriz 28×28. Claramente, o parámetro de fiestra deberá ser moito máis pequeno nas variables do centro da imaxe que nas que correspondan aos bordes.

En resumo, os métodos clásicos de estimación da densidade teñen graves problemas computacionais segundo aumenta a dimensión, ademais doutros problemas de índole práctica e teórica, como a necesidade de máis parámetros na estimación tipo núcleo. Este feito motiva a busca de novas aproximacións para modelar datos de alta dimensión, que será o obxecto dos Capítulos 2 e 3.

1.2. Medidas de axuste do modelo

Dende un punto de vista experimental, necesitaremos ferramentas que nos permitan cuantificar o grao de axuste dun modelo cunha mostra, así como metodoloxías sistemáticas para comparar diferentes modelos entre si. Esta sección presentará, con este fin, varios conceptos da Estatística e da Teoría da Información.

1.2.1. A función de verosimilitude

A estimación por técnicas de máxima verosimilitude é unha ferramenta fundamental da inferencia paramétrica, e o concepto resulta de gran aplicación noutros contextos. Para introducila, consideremos unha mostra aleatoria simple de n observacións:

$$\boldsymbol{X} = (X_1, \dots, X_n),$$

onde as observacións X_i son independentes e seguen a mesma distribución dada pola función de densidade f, que é descoñecida. Sexa $\mathcal{F}_\theta = \{f_\theta, \theta \in \Theta\}$ unha familia paramétrica de distribucións de probabilidade candidatas a aproximar f. A estimación por máxima verosimilitude, esencialmente, propón un cambio de vista sobre a función de densidade, entendéndoa como unha función do parámetro no canto de ser unha función dos datos. Deste xeito, para uns datos dados, optimízase e considerase o parámetro que maximice a función de densidade, é dicir, que sexa máis verosímil:

$$\hat{\theta} = \arg\max_\theta f_\theta(\boldsymbol{X}) = \arg\max_\theta \prod_{i=1}^{n} f_\theta\left(X_i\right).$$

A función $\mathcal{L}(\theta) = f_\theta(X)$ denomínase función de verosimilitude. Como o problema de optimización non cambia ao introducir un logaritmo, unha formulación alternativa moi empregada e vantaxosa para os cálculos é:

$$\hat{\theta} = \arg\max_\theta \sum_{i=1}^{n} \log f_\theta\left(X_i\right). \tag{1.2}$$

A maximización da función de verosimilitude respecto á mostra pódese facer analiticamente ou numericamente. Nos próximos capítulos, todos os modelos presentados farán unha maximización numérica empregando o gradiente da función de verosimilitude.

1.2.2. Teoría da Información: entropía, diverxencia de Kullback–Leibler e entropía cruzada

A diverxencia de Kullback–Leibler proporciona unha medida de como de diferentes son dúas distribucións dadas sobre a mesma variable aleatoria. O seu uso provén da Teoría da Información, unha disciplina fundada por Claude E. Shannon e Warren Weaver a finais dos anos 40 do século XX que estuda principalmente a transmisión de sinais sobre medios ruidosos. Pese a partir dun obxectivo moi especializado, as matemáticas construídas neste marco formalizan conceptos aplicables en moitos eidos da Estatística e das Ciencias da Computación. Polo tanto, a continuación introduciremos os fundamentos deste campo.

A Teoría da Información fundaméntase na idea de que eventos poucos probables conteñen moita información, mentres que eventos seguros non proporcionan información ningunha. Esta

intuición formalízase no concepto de autoinformación dun suceso $x \in \mathcal{X}$ (sendo $\mathcal{X}$ o espazo de sucesos), que se define como:

$$I(x) = -\log \mathbb{P}(x).$$

É dicir, a autoinformación tende a infinito cando a probabilidade tende a cero, e é nula cando $\mathbb{P}(x) = 1$. Pasando ao espazo de probabilidade, supoñamos agora que x é un dato de X, unha variable aleatoria con soporte real e función de densidade f_X. A cantidade de información que esperamos obter en media dun evento extraído da correspondente distribución denomínase *entropía de Shannon* da distribución, e defínese como:

$$\mathcal{H}(X) = \mathbb{E}_X[I(X)] = -\mathbb{E}_X[\log f_X(X)] = -\int_{\mathbb{R}} f_X(u) \log f_X(u)\ du.$$

Tal e como suxire o nome, as distribucións que concentran toda a probabilidade nunha rexión pequena e están preto de ser deterministas teñen unha baixa entropía. Pola contra, as distribucións que se aproximen á distribución uniforme terán unha entropía moi alta.

Estamos en condicións de definir a diverxencia de Kullback–Leibler. Supoñamos que X e Y son dúas variables aleatorias co mesmo soporte real e funcións de densidade asociadas f_X e g_Y.

$$D_{\mathrm{KL}}(X\|Y) = \mathbb{E}_X\left[\log \frac{f_X(X)}{g_Y(X)}\right] = \mathbb{E}_X[\log f_X(X) - \log g_Y(X)].$$

Pódese interpretar, para unha variable aleatoria X, como a cantidade media de información extra ao usar a distribución de Y para explicar X. Recordemos que "información extra" significa que os eventos son menos probables, polo que canto máis alta é a diverxencia, máis diferentes son as distribucións de Y e X. Por este motivo, resulta moi útil para comparar distribucións, e ten varias propiedades interesantes: é non negativa, será cero se as distribucións son iguais e non é simétrica. Esta última propiedade imposibilita que sexa unha métrica entre distribucións.

Pódese atopar unha visión completa da Teoría da Información en Cover e Thomas (2006). Nos próximos Capítulos, empregaremos as propiedades da diverxencia KL como unha pseudo-distancia que nos permita axustar unha distribución para que se achegue a outra.

A estimación por máxima verosimilitude minimiza a D_{KL}

Fixémonos nunha das conexións máis evidentes coa Inferencia Estatística. Sexa $\boldsymbol{X} = (X_1, \dots, X_n)$ unha mostra aleatoria simple da variable aleatoria X, con soporte real e densidade descoñecida f_X. Consideremos a familia paramétrica $\mathcal{F}_\theta = \{f_\theta, \theta \in \Theta\}$, e sexan X_θ as variables aleatorias asociadas á mesma. Por simplicidade, supoñamos que $f_X = f_{\theta_0}$, para un θ_0 que pretendemos estimar en $\hat{\theta}$. A estimación por máxima verosimilitude garante que obteremos o $\hat{\theta}$ tal que $f_{\hat{\theta}}$ é a

máis verosímil baixo $\mathcal{F}_\theta$, aínda que non se cumpra que $f_X \in \mathcal{F}_\theta$.

$$\begin{aligned}\hat{\theta} &= \arg\max_\theta \mathcal{L}(\theta) = \arg\max_\theta f_\theta(\boldsymbol{X}) = \arg\max_\theta \prod_{i=1}^n f_\theta(X_i) = \arg\max_\theta \sum_{i=1}^n \log f_\theta(X_i) \\ &= \arg\max_\theta \left(\sum_{i=1}^n \log f_\theta(X_i) - \sum_{i=1}^n \log f_X(X_i)\right) = \arg\max_\theta \sum_{i=1}^n (\log f_\theta(X_i) - \log f_X(X_i)) \\ &= \arg\max_\theta \sum_{i=1}^n \log \frac{f_\theta(X_i)}{f_X(X_i)} = \arg\min_\theta \sum_{i=1}^n \log \frac{f_X(X_i)}{f_\theta(X_i)} = \arg\min_\theta \frac{1}{n} \sum_{i=1}^n \log \frac{f_X(X_i)}{f_\theta(X_i)}.\end{aligned}$$

Sexa agora $h_\theta(x) = \log \frac{f_X(x)}{f_\theta(x)}$ e apliquemos a lei dos grandes números:

$$\begin{aligned}&\hat{\theta} = \arg\min_\theta \frac{1}{n} \sum_{i=1}^n h_\theta(X_i) \quad \underset{n\to\infty}{\longrightarrow} \quad \arg\min_\theta \mathbb{E}_X[h_\theta(X)] \\ &\hat{\theta} \underset{n\to\infty}{\longrightarrow} \arg\min_\theta \mathbb{E}_X\left[\log \frac{f_X(X)}{f_\theta(X)}\right] = \arg\min_\theta D_{\mathrm{KL}}(X \parallel X_\theta).\end{aligned}$$

É dicir, para unha mostra suficientemente grande, a estimación por máxima verosimilitude minimiza a diverxencia de Kullback–Leibler da familia paramétrica respecto á distribución real descoñecida.

Finalmente, definamos un concepto moi relacionado coa diverxencia KL, a entropía cruzada de dúas distribucións. Sexan X e Y dúas variables aleatorias con soporte real e funcións de densidade asociadas f_X e g_Y,

$$\mathcal{H}(X, Y) = \mathcal{H}(X) + D_{\mathrm{KL}}(X \parallel Y) = -\mathbb{E}_X[\log g_Y(X)] = -\int_{\mathbb{R}} f_X(u) \log g_Y(u)\, du.$$

1.3. Aprendizaxe automática e redes neuronais

A Estatística aborda a modelización de fenómenos estocásticos de xeito rigoroso, tendo presentes todas as garantías teóricas. A aprendizaxe automática, pola contra, é unha disciplina da Intelixencia Artificial e das Ciencias da Computación que pese a ter obxectivos semellantes, admite unha relaxación das hipóteses e cede o protagonismo á computación masiva e aos datos.

En liñas xerais, a aprendizaxe automática desenvolve técnicas e algoritmos que aproveitan a experiencia para resolver problemas, cunha aproximación eminentemente práctica. Nesta sección presentaremos as redes neuronais, que constitúen o exemplo por excelencia deste tipo de algoritmos e que serán empregadas no Capítulo 3 por NADE.

En esencia, as redes neuronais son un modelo computacional moi simple e flexible que aproxima funcións arbitrarias en base a unha colección de exemplos. A súa popularidade e difusión creceron enormemente nos últimos anos debido, por unha parte, a que posibilitan explotar as

inxentes cantidades de computación e de datos dispoñibles actualmente e, por outra, á capacidade que posúen para capturar patróns moi abstractos sobre os datos. O representante máis salientable desta familia de algoritmos é o perceptrón multicapa, tamén chamado rede neuronal *feedforward*.

Denomínanse redes neuronais por inspirarse vagamente no sistema nervioso, sen pretender modelar o funcionamento do cerebro humano. O nome "rede *feedforward*" é moito máis explicativo do seu funcionamento real: a información flúe cara adiante a través da composición de varias funcións simples que van transformando os datos. Estas operacións simples teñen parámetros que son aprendidos nun proceso denominado adestramento. A forza deste paradigma reside na simpleza da parametrización, que posibilita implementacións cun custo computacional moi afinado sobre procesadores de propósito específico, como tarxetas gráficas.

No plano teórico, probouse analiticamente que dada unha función $f : \mathbb{R}^n \to \mathbb{R}^m$ arbitraria, existe unha rede *feedforward* $F : \mathbb{R}^n \to \mathbb{R}^m$ que se achega tanto como sexa necesario na norma do supremo, dado un ancho ou profundidade da rede suficientemente grandes. Este resultado denomínase Teorema de Aproximación Universal e existen diversas variantes do teorema segundo o tipo de rede (das cales Pinkus (1999) é a máis coñecida).

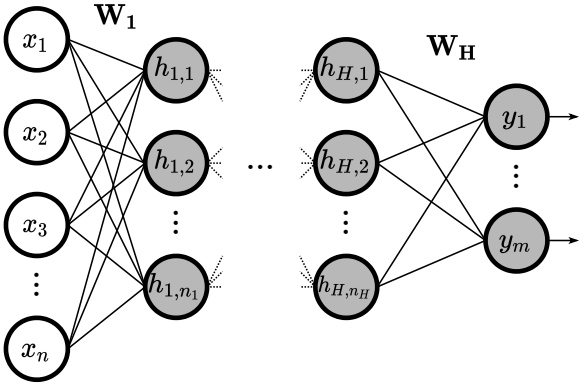


Figura 1.2: Modelo básico dunha rede neuronal *feedforward*. As x denotan ás entradas, as h ás capas ocultas e as y á capa de saída. As conexións entre capas denótanse mediante as matrices de pesos $\boldsymbol{W_i}$.

A Figura 1.2 ilustra a estrutura básica dunha rede *feedforward*. A rede da figura ten n entradas, H capas ocultas de n_i neuronas (n_i varía coa capa $i = 1, \dots, H$) e m neuronas na capa de saída. Dise que as redes son "profundas" cando teñen máis dunha capa oculta.

Matematicamente, unha rede é unha aplicación continua e diferenciable $F : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ composta dunha sucesión de capas. Cada capa $\boldsymbol{h}_i$ opera sobre os resultados da capa anterior $\boldsymbol{h}_{i-1}$, aplicando unha transformación lineal seguida dunha función de activación f_{act}. Especificamente,

cada neurona $h_{i,j}$ realiza unha suma ponderada das neuronas da capa anterior $h_{i-1,j}$ (máis un termo fixo) e ao resultado aplícalle a función de activación non linear. É dicir:

$$h_{i,j} = f_{\text{act}}\left(b + \sum_{l=1}^{n_{i-1}} w_l h_{i-1,l}\right).$$

Os parámetros a aprender serán $(b, w_1, \dots, w_{n_{i-1}})$, distintos para cada neurona. Unha elección común para a función de activación f_{act} é a función sigmoide, $\sigma(x) = \frac{1}{1+e^{-x}}$. Se non se aplicase a función de activación e todas as neuronas fosen unha combinación linear das entradas, teríamos que, ao ser linear a composición de aplicacións lineares, poderíanse substituír todas as capas ocultas da rede por unha ponderación particular das entradas na capa de saída. Isto implicaría que a rede non tería capacidade para modelar funcións non lineares.

As redes *feedforward* admiten unha sinxela formulación matricial. No exemplo da Figura 1.2, sexa $\boldsymbol{W_i}$ a matriz cuxa fila j-ésima ten os pesos w para as entradas da neurona $h_{i,j}$. Sexa tamén $\boldsymbol{x} = (x_1, \dots, x_n)'$, $\boldsymbol{y} = (y_1, \dots, y_m)'$, $\boldsymbol{h}_i = (h_{i,1}, \dots, h_{i,n_i})'$ e $\boldsymbol{b}_i$ o vector de termos independentes para cada unha das neuronas da capa i. Pódese escribir:

$$\begin{aligned}
\boldsymbol{h}_1 &= f_{\text{act}}(\boldsymbol{b}_1 + \boldsymbol{W}_1\boldsymbol{x}),\\
\boldsymbol{h}_i &= f_{\text{act}}(\boldsymbol{b}_{i-1} + \boldsymbol{W}_{i-1}\boldsymbol{h}_{i-1}), \quad i \in \{2, \dots, H\}\\
\boldsymbol{y} &= f_{\text{act}}(\boldsymbol{b}_H + \boldsymbol{W}_H\boldsymbol{h}_H).
\end{aligned}$$

O axuste dos parámetros nas redes neuronais adoita realizarse a través do algoritmo iterativo coñecido como *backpropagation*. En primeiro lugar, compútase unha función de perda para os exemplos de adestramento. En segundo lugar, o algoritmo calcula recursivamente o gradiente da función de perda respecto a cada parámetro da rede. Finalmente, axústanse os parámetros proporcionalmente, e este proceso repítese seguindo un algoritmo de descenso do gradiente. A función de perda adoita estar baseada na función de verosimilitude. Todos os aspectos das redes neuronais e o seu adestramento desenvólvense con detalle en Goodfellow et al. (2016).

1.4. Un exemplo de xoguete: o modelo LARC

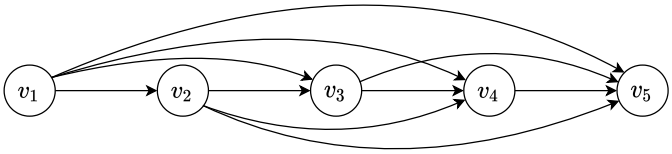
Frey (1998) realizou un experimento que resultou ser crucial para desenvolvementos posteriores. Tomou unha versión do conxunto de datos MNIST reducida a 11.000 imaxes, 64 píxeles por imaxe (8×8) e valores binarios en cada pixel. Neste conxunto de probas, preparou diversos modelos gráficos de estimación de densidade, baseados nas ideas que se desenvolven no Capítulo 2. Comparou estes modelos medindo a súa utilidade para clasificar o díxito visible na imaxe, e os resultados foron sorprendentes: o mellor modelo resultou ser, por unha ampla marxe, o máis

simple e menos custoso computacionalmente. A efectividade e eficiencia de LARC (do inglés *Logistic Autoregressive Classifier*) para modelar datos semellantes ao MNIST foi unha influencia clave para o desenvolvemento do modelo que se estuda no Capítulo 3: NADE.

O exemplo de LARC serviranos para contextualizar os desenvolvementos teóricos presentados nos próximos capítulos. Formalmente, unha imaxe desta versión reducida do MNIST é un vector $\boldsymbol{v} = (v_1, \dots, v_{64}) \in \{0,1\}^{64}$, onde un pixel en negro represéntase con un 1 e en branco con un 0:

v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}	v_{16}
v_{17}	v_{18}	v_{19}	v_{20}	v_{21}	v_{22}	v_{23}	v_{24}
v_{25}	v_{26}	v_{27}	v_{28}	v_{29}	v_{30}	v_{31}	v_{32}
v_{33}	v_{34}	v_{35}	v_{36}	v_{37}	v_{38}	v_{39}	v_{40}
v_{41}	v_{42}	v_{43}	v_{44}	v_{45}	v_{46}	v_{47}	v_{48}
v_{49}	v_{50}	v_{51}	v_{52}	v_{53}	v_{54}	v_{55}	v_{56}
v_{57}	v_{58}	v_{59}	v_{60}	v_{61}	v_{62}	v_{63}	v_{64}

(a)



(b)

Figura 1.3: Na esquerda, exemplo de imaxe 8×8 binarizada correspondente a un díxito do número 0 manuscrito. Na dereita, esquema do modelo LARC.

LARC modela a imaxe dun xeito autorregresivo sobre as variables binarias v_i, factorizando a función masa de probabilidade sobre a secuencia de variables. Por simplicidade da notación, f denotará á función masa de probabilidade:

$$f(\boldsymbol{v}) = \prod_{i=1}^{n} f\left(v_i \mid \{v_k\}_{k=1}^{i-1}\right) = f(v_1) \cdot f(v_2 \mid v_1) \cdot f(v_3 \mid v_2, v_1) \cdot \dots$$

A Figura 1.3b ilustra a natureza autorregresiva do modelo: cada nova variable depende de todas as anteriores. De momento, esta factorización pode parecer axeitada para datos secuenciais (como series temporais) pero moi arbitraria para imaxes, cuxos píxeles admiten varias ordenacións igualmente válidas. No Capítulo 3, verase que esta escolla vén dada pola aplicación dun dos modelos gráficos máis simples.

Finalmente vexamos como estima LARC as compoñentes $f\left(v_i \mid \{v_k\}_{k=1}^{i-1}\right)$. Como indica o seu nome, utiliza unha regresión loxística, estimando a variable v_i en función das $i-1$ variables anteriores:

$$f\left(v_i \mid \{v_k\}_{k=1}^{i-1}, \boldsymbol{\theta}_i\right) = v_i \; \sigma\left(\sum_{k=0}^{i-1} \theta_{i,k} v_k\right) + (1 - v_i)\left[1 - \sigma\left(\sum_{k=0}^{i-1} \theta_{i,k} v_k\right)\right],$$

sendo $\sigma(x) = \frac{1}{1+e^{-x}}$ a función sigmoide, $v_0 = 1$ unha variable ficticia e $\boldsymbol{\theta}_i = (\theta_{i,1}, ..., \theta_{i,i-1})$ o parámetro que pondera as $i-1$ variables anteriores na estimación de v_i. Ao longo deste traballo, $\boldsymbol{\theta}$ usarase para denotar o conxunto de parámetros dos diferentes modelos.

Frey (1998) adestrou os parámetros utilizando un algoritmo baseado en máxima verosimilitude (pero impoñendo certas condicións sobre os parámetros). O rendemento foi moi superior ao resto de modelos da comparativa, suxerindo que esta aproximación tan sinxela captura moi ben a densidade desta clase de distribucións, e salientando as posibilidades que ofrecen os modelos gráficos.

O modelo LARC, pese aos bos resultados obtidos, ten dúas limitacións principais. Por unha parte, ao utilizar a regresión loxística, non ten capacidade para modelar unha resposta non lineal ás entradas. Esta baixa capacidade impide que sexa útil para outras tarefas como a de simular mostras. Por outra parte, LARC non é moi eficiente en canto á parametrización xa que o número de parámetros, que é $(n^2 - n)/2$, medra moi rapidamente respecto ao número de variables.

A maior cualidade de LARC é a relación entre rendemento e custo computacional. A simplicidade do modelo permite que a estimación da densidade sexa moi rápida, así como a súa aplicación en tarefas de clasificación. No Capítulo 2 veremos o paradigma oposto: a máquina de Boltzmann restrinxida (Sección 2.1.7) terá moita capacidade para modelar dependencias complexas, a cambio de ser un modelo moi dificultoso de operar. O caso de estudo deste traballo, NADE (Capítulo 3) intentará xuntar o mellor dos dous mundos: un modelo sinxelo como LARC pero con moita capacidade de modelado.

Capítulo 2

Modelos gráficos

As distribucións semellantes ao conxunto de datos do MNIST, que foi presentado na introdución, involucran un número moi grande de variables aleatorias. Neste caso, cada un dos 768 píxeles é unha variable que depende das 767 restantes. Describir estas distribucións mediante unha única función de densidade $f : \mathbb{R}^n \longrightarrow \mathbb{R}^+$ é moi complexo e presenta un espazo de busca inabarcable para o proceso de estimación. Con todo, podemos observar que o habitual nestes contextos de alta dimensión é que cada variable interactúe só cun subconxunto pequeno do resto de variables, de xeito que poidamos desprezar a gran maioría de posibles interaccións. No caso dos díxitos do MNIST, intuitivamente sabemos que hai unha dependencia moi forte entre as variables correspondentes a píxeles próximos espacialmente, e moita menos relación entre variables de píxeles afastados na imaxe. Parece natural representar estas dependencias complexas entre variables empregando a Teoría de Grafos.

Os modelos gráficos empregan esta intuición para construír distribucións a partir do grafo de relacións entre variables aleatorias. Para introducilos, consideremos tres variables aleatorias reais X_1, X_2 e X_3, con cadansúa función de densidade que denotamos como f_1, f_2 e f_3. Baixo a hipótese de que non todas se inflúen simultaneamente entre si, supoñamos que X_1 inflúe sobre X_2 e X_2, á súa vez, inflúe a X_3, seguindo a estrutura da Figura 2.1. Dado que X_1 condiciona a X_2, interésanos a densidade condicional $f_2(\cdot \mid X_1 = x_1)$.

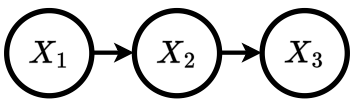


Figura 2.1: Grafo correspondente ás relacións entre variables do exemplo.

Neste suposto, a función de densidade da distribución conxunta para unha observación

(x_1, x_2, x_3) pódese factorizar do seguinte xeito:

$$f(x_1, x_2, x_3) = f_1(x_1) f_2(x_2 \mid X_1 = x_1) f_3(x_3 \mid X_2 = x_2). \tag{2.1}$$

Realizar esta simplificación en factores ten varias vantaxes. En primeiro lugar, reduce enormemente o número de parámetros necesarios para describir a distribución, que adoita medrar exponencialmente co número de variables. En segundo lugar, permite estruturar a modelización de problemas complexos, creando un marco teórico común que separa a análise da dependencia entre variables, a modelización dos distintos factores e os métodos de inferencia.

As dependencias entre as variables que determinan a factorización poden ser moi complexas, e a Teoría de Grafos proporciona unha representación moi axeitada, como a da Figura 2.1. Debido xustamente a esta representación mediante grafos, esta clase de modelos son coñecidos como *modelos probabilísticos estruturados* ou *modelos gráficos*.

Recordemos que un grafo é un conxunto de nodos ou vértices conectados mediante enlaces, que reciben o nome de arestas. Un grafo dirixido é aquel cuxas arestas posúen un sentido, marcando un nodo de saída e outro de entrada. Pola contra, nun grafo non dirixido as arestas conectan nodos sen indicar un sentido no enlace.

Segundo o tipo de grafo que se empregue na representación, os modelos gráficos tamén se clasifican en dirixidos ou non dirixidos. O feito de que exista un condicionamento dirixido entre variables aleatorias ten moitas implicacións no modelado e na inferencia, provocando unha fonda separación nos métodos empregados en cada clase. Pese a todo, é unha diferenza soamente na representación escollida para modelar as interaccións entre o conxunto de variables aleatorias, e existen técnicas para converter certas representacións non dirixidas en dirixidas e viceversa. Para algúns problemas, pode que encaixen os dous enfoques pero, en xeral, a escolla da representación dependerá das características do problema e de cal resulte máis natural.

A continuación, estudaremos en profundidade as dúas clases de modelos. Antes, formulemos unha definición xeral dos modelos gráficos, inspirada por Jordan (2004) e que resume especialmente ben a esencia destes modelos.

Un modelo gráfico é unha familia de distribucións de probabilidade caracterizada a través dun grafo. Os nodos do grafo correspóndense con variables aleatorias, que definen unha distribución conxunta do modelo mediante produtos de funcións sobre subconxuntos conectados do grafo. O énfase desta representación na estrutura de grafo permite formular algoritmos xerais para o cómputo de distribucións marxinais ou condicionais, así como para o axuste paramétrico dos modelos. As vantaxes de manter esta estrutura son, por unha parte, un férreo control do custo computacional das operacións e, por outra, a naturalidade dos grafos para modelar moitos problemas que xorden noutros campos.

2.1. Modelos non dirixidos ou campos aleatorios de Markov

Os campos aleatorios de Markov xorden do estudo de certos procesos físicos estocásticos. Denomínanse "campos aleatorios" por ser un proceso estocástico sobre un soporte multivariante e "de Markov" por cumprir unha noción de localidade na dependencia entre variables, ao ser estas dependencias exclusivamente entre veciños, como veremos na Sección 2.1.4. Un dos primeiros exemplos foi a exposición a un campo magnético dun material ferromagnético, que se modela como unha cuadrícula ou cadea de estados na que cada nodo inflúe nos seus veciños, sendo esta influencia a mesma nos dous sentidos. A representación como grafo non dirixido é moi natural para esta clase de problemas.

Porén, modelar condicionamentos bidireccionais entre variables é máis complexo que o caso dirixido. O problema fundamental xorde, de xeito intuitivo, ao intentar escribir a función de densidade do mesmo xeito que fixemos no primeiro exemplo (ecuación (2.1)). Vemos que, debido a que as arestas xa non teñen un sentido, non podemos empregar un produto de densidades condicionais, e esta diferenza fundamental provoca que as ferramentas para realizar inferencia e estimar a densidade muden por completo.

Para introducir estas novas ferramentas, empregaremos a Figura 2.2, que amosa un novo exemplo de modelo non dirixido de variables aleatorias reais. Antes, recordemos un concepto moi sinxelo da teoría de grafos que será fundamental para o que segue.

Nun grafo non dirixido $\mathcal{G}$, un *clan* (do inglés *clique*) $\mathcal{C}$ é un subconxunto de vértices de $\mathcal{G}$ que cumpre a seguinte condición: todo vértice de $\mathcal{C}$ está conectado co resto de vértices de $\mathcal{C}$, e non se poden engadir máis vértices a $\mathcal{C}$ sen romper esta propiedade. Na Figura 2.2, as caixas delimitan os clans do grafo: $\mathcal{C}_1(\boldsymbol{X}) = (X_1, X_2, X_3)$, $\mathcal{C}_2(\boldsymbol{X}) = (X_2, X_4)$ e $\mathcal{C}_3(\boldsymbol{X}) = (X_3, X_5)$.

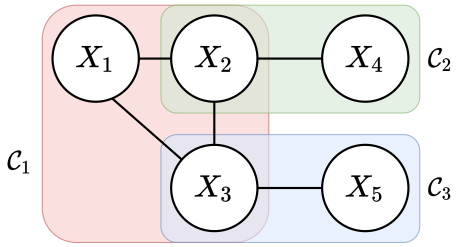


Figura 2.2: Exemplo de modelo gráfico non dirixido con tres clans: $\mathcal{C}_1$, $\mathcal{C}_2$ e $\mathcal{C}_3$.

A parametrización dun modelo non dirixido realízase empregando estes clans. Cada clan do grafo ten asociada un factor ϕ_i, denominado *potencial do clan*, e o produto dos potenciais dos clans constitúe a densidade non normalizada:

$$\tilde{f}(\boldsymbol{x}) = \prod_{i=1}^{C} \phi_i \left(\mathcal{C}_i \left(\boldsymbol{x}\right)\right), \tag{2.2}$$

onde C é o número de clans do grafo. Pese a que $\phi_i, i = 1, \dots, C$ son funcións non negativas das variables aleatorias do clan, o seu produto non ten que integrar 1 no soporte necesariamente. Este é o motivo da introdución dunha constante de normalización, que na literatura se denota habitualmente como Z, e permite escribir a función de densidade:

$$f(\boldsymbol{x}) = \frac{1}{Z}\tilde{f}(\boldsymbol{x}), \text{ sendo } Z = \int_{\mathbb{R}^n} \tilde{f}(\boldsymbol{u})d\boldsymbol{u}. \tag{2.3}$$

Z soe ser moi custosa computacionalmente de calcular polo que, habitualmente, os factores deseñaranse de xeito que faciliten o cálculo ou a aproximación de Z.

2.1.1. Modelos baseados en enerxía ou máquinas de Boltzmann

Os modelos non dirixidos presentados teñen varias dificultades prácticas, ao intentar usalos para aproximar a distribución da que proveñen os datos. Ademais do problema de calcular a constante de normalización fZ, se formulamos a busca dos ϕ_i da ecuación (2.2) como un problema de optimización, a busca debe ser restrinxida a funcións non negativas e integrables no soporte, o cal complica aínda máis este problema de optimización.

Estas dificultades motivaron a introdución de variantes máis simples, como os modelos baseados en enerxía, tamén chamados máquinas de Boltzmann. Estas variantes definen a densidade non normalizada mediante unha función de enerxía $E : \mathbb{R}^n \longrightarrow \mathbb{R}$:

$$\tilde{f}(\boldsymbol{x}) = \exp\{-E(\boldsymbol{x})\}.$$

Deste xeito, queda garantida a non negatividade de $\tilde{f}$ independentemente da escolla da función de enerxía. Ademais, tamén resulta moi oportuno que os sumandos da función de enerxía serán factores de $\tilde{f}$, de xeito que se pode deseñar a través dos sumandos de E os potenciais da ecuación (2.2).

2.1.2. Modelado e axuste das dependencias entre variables

Á hora de empregar estes modelos para abordar un problema real, debemos planificar tres aspectos moi importantes: atopar unha estrutura para a rede que sexa adecuada, deseñar como imos modelar os factores individuais da rede e, por último, seleccionar métodos de estimación de parámetros e de inferencia admisibles computacionalmente que permitan extraer proveito e conclusións do modelo.

Fixémonos no primeiro problema: fixar unha estrutura da rede que modele as dependencias. No principio do Capítulo, xa adiantamos que o obxectivo será modelar as interaccións máis importantes e desprezar o resto. Porén, pese a este suposto, o habitual é que exista unha forte

relación entre moitas das variables observadas. Isto é un problema para os modelos gráficos: cantas máis relacións, máis grandes serán os clans e menos elementos terá a factorización da función de densidade.

Para solucionar este problema, moitos modelos optan por empregar variables latentes. Estas variables poden existir de forma natural no problema modelado (como era o caso cando as introducimos na Sección 1.1.2) ou pódese supoñer a súa existencia, fixando un grafo independente do problema (como é o caso da máquina de Boltzmann da Sección 2.1.7). Denotemos estas variables por $\boldsymbol{H} = (H_1, H_2, \dots, H_m)$, de xeito que as variables do modelo, que denotaremos por $\boldsymbol{X}$, descompoñeranse en visibles ($\boldsymbol{V}$) e latentes ($\boldsymbol{H}$):

$$\boldsymbol{X} = (\boldsymbol{V}, \boldsymbol{H}) = (V_1, V_2, \dots, V_n, H_1, H_2, \dots, H_m).$$

A introdución de variables latentes permite, sen considerar clans moi grandes, que os modelos capturen dependencias entre calquera dúas variables visibles V_i e V_j, a través das dependencias de V_i con $\boldsymbol{H}$ e de $\boldsymbol{H}$ con V_j. Esta aproximación é moi proveitosa na práctica: é máis barato computacionalmente incrementar o número de variables do modelo que ter unha factorización da función de densidade con menos elementos.

Outra vantaxe das variables latentes é que proporcionan unha alternativa de deseño moi interesante: por un lado, pódense escoller coidadosamente as variables segundo as relacións que identifique un experto nos datos (dándolle un significado a cada variable latente, como fixemos na Sección 1.1.2) ou, alternativamente, pódese fixar unha estrutura xenérica de variables latentes que sirva para moitos problemas e delegue o modelado de dependencias na estimación de parámetros da rede (como fai a máquina de Boltzmann restrinxida da Sección 2.1.7).

Agora, os modelos con variables latentes estimarán

$$f : \boldsymbol{x} = (\boldsymbol{v}, \boldsymbol{h}) \in \mathbb{R}^{n+m} \to f(\boldsymbol{x}) \in \mathbb{R}^{+}.$$

De primeiras, pode parecer un paso atrás, ao depender dun conxunto de variables aleatorias $\boldsymbol{H}$ que non podemos observar. A función que si podemos observar,

$$f_{\boldsymbol{V}}(\boldsymbol{v}) = \int_{\mathbb{R}^m} f(\boldsymbol{v}, \boldsymbol{h}) d\boldsymbol{h},$$

agora necesita do cálculo da distribución marxinal sobre $\boldsymbol{H}$, no que parece unha integral intratable computacionalmente. Na seguinte sección, veremos como resolver este problema e na Sección 2.1.1, ilustrarase a intratabilidade deste tipo de integrais mediante un exemplo.

Finalmente, un atractivo adicional das variables latentes é que proporcionan unha representación alternativa das variables visibles. Un modelo que estime ben a variable "compoñente da

mestura" da Sección 1.1.2 pode ser moi útil para clasificación de mostras nunha ou noutra compoñente. Un modelo que estime a densidade da distribución correspondente ao MNIST podería aprender unha variable latente que se corresponda co díxito que representa cada mostra. Incluso no caso de que as variables latentes non se deseñaran especificamente e foran aprendidas nun proceso de estimación paramétrica, estas variables conteñen información das interaccións entre variables moi útil para facer inferencia sobre o modelo.

2.1.3. Os problemas da inferencia exacta e da simulación

O obxectivo de construír estes modelos é poder obter conclusións sobre a poboación da que obtivemos as mostras, o que se denomina inferencia estatística. Como vimos, moitos dos parámetros interesantes da poboación van estar relacionados coas variables aleatorias latentes $\boldsymbol{H}$, polo que será de especial interese estimar o valor desas variables dadas as visibles, é dicir, computar a esperanza

$$\mathbb{E}_{\boldsymbol{H}}[\boldsymbol{H} \mid \boldsymbol{V} = \boldsymbol{v}] = \int_{\mathbb{R}^m} \boldsymbol{h} f(\boldsymbol{h} \mid \boldsymbol{v}) d\boldsymbol{h}, \text{ sendo } f(\boldsymbol{h} \mid \boldsymbol{v}) = \frac{f(\boldsymbol{v}, \boldsymbol{h})}{f_{\boldsymbol{V}}(\boldsymbol{v})} = \frac{f(\boldsymbol{v}, \boldsymbol{h})}{\int_{\mathbb{R}^m} f(\boldsymbol{v}, \boldsymbol{h}) d\boldsymbol{h}}. \tag{2.4}$$

Computar densidades condicionais é un dos problemas que máis xorden ao realizar inferencia, e acabamos de comprobar como computar condicionais implica resolver integrais de distribucións marxinais sobre algunhas das variables. A necesidade de computar condicionais vai moito máis alá: se queremos estimar por máxima verosimilitude uns parámetros do modelo, debemos calcular a densidade marxinal $f_{\boldsymbol{V}}$ (recordemos que os valores de $\boldsymbol{H}$ son descoñecidos nunha mostra), tal como fixemos na ecuación (2.4) ou, de xeito equivalente, despexando a densidade marxinal a partir da conxunta e da condicional.

En conclusión, facer inferencia seguindo unha aproximación exacta como a da ecuación (2.4) é intratable computacionalmente. Aínda no caso de traballar con variables discretas, segue a ser computacionalmente intratable debido ao crecemento exponencial das posibles combinacións dos valores das variables latentes. Nas seguintes seccións, exploraremos algunhas aproximacións que si son computables, realizando inferencia aproximada nestes modelos.

Finalmente, outra tarefa relacionada coa inferencia, de gran interese unha vez construído un modelo, é simular mostras do mesmo. Nun modelo dirixido como o exemplo da Figura 2.1 do comezo do Capítulo, é sinxelo: simulamos unha observación x_1 de X_1, logo unha observación x_2 de $X_2 \,|\, X_1 = x_1$ e finalmente unha observación x_3 de $X_3 \,|\, X_2 = x_2$, de xeito que obtemos (x_1, x_2, x_3). Nun modelo gráfico con ciclos ou non dirixido coma o da Figura 2.2, non é posible seguir esta aproximación. Ao non existir unha ordenación natural clara das variables para obter as mostras na simulación, preséntasenos o problema infinito do ovo e a galiña, que non ten unha solución clara. Nas seguintes seccións, exploraremos maneiras de sortear este problema e veremos

como a simulación de mostras tamén posibilita algunhas formas de inferencia aproximada.

2.1.4. Métodos Monte Carlo e mostraxe de Gibbs

A sección anterior ilustra como moitos problemas de inferencia requiren do cálculo de integrais ou de sumas sobre distribucións marxinais, intratables para modelos de gran tamaño. Non é o único caso no que se debe calcular este tipo de integrais. Por exemplo, o cómputo da constante de normalización Z da ecuación (2.3) tamén é un problema análogo. A maiores, desexamos que un modelo útil sexa capaz de simular mostras, sen a restrición previa de ser dirixidos e non cíclicos.

A mostraxe de Gibbs e os métodos Monte Carlo pretenden arranxar estes problemas proporcionando unha solución aproximada. O método de mostraxe de Gibbs é un algoritmo iterativo que permite simular mostras evitando o problema do ovo e a galiña e, pola súa parte, os métodos Monte Carlo empregan a simulación de mostras para estimar moitas destas sumas ou integrais que resultaban intratables computacionalmente. Koller e Friedman (2009) desenvolven estas técnicas e analizan as particularidades da súa aplicación sobre modelos gráficos.

Métodos Monte Carlo

Os métodos de Monte Carlo aproximan sumas ou integrais difíciles de computar de forma exacta, intercambiando exactitude no cálculo por velocidade. Intuitivamente, consisten en formular a suma ou integral desexada como unha esperanza baixo unha distribución da que saibamos obter mostras, de xeito que a media mostral aproxime a esperanza que buscamos.

Formalmente, no caso dunha integral intratable $\int h(x)dx$, buscaremos atopar unha función g e unha variable aleatoria X con densidade f de xeito que $h = f \cdot g$, o que nos permite reescribir

$$s = \int h(x)dx = \int f(x)g(x)dx = \mathbb{E}_X\left[g(X)\right].$$

Dada unha mostra aleatoria simple $\boldsymbol{X} = (X_1, \dots, X_n)$, o estimador de s será

$$\hat{s}_n = \frac{1}{n}\sum_{i=1}^{n} g(X_i).$$

Este estimador, por construción, non ten nesgo, xa que

$$\mathbb{E}\left[\hat{s}_n\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[g\left(X_i\right)\right] = \frac{1}{n}\sum_{i=1}^{n} s = s$$

e a súa varianza diminúe co tamaño da mostra (a taxa paramétrica):

$$\mathrm{Var}\left[\hat{s}_n\right] = \frac{1}{n^2}\sum_{i=1}^{n}\mathrm{Var}[g(X_i)] = \frac{\mathrm{Var}[g(X)]}{n}.$$

A lei dos grandes números garántenos que o estimador converxe de forma case segura a s e, grazas ao teorema central do límite, sabemos que a distribución de $\hat{s}_n$ converxe a unha normal de media s e varianza $\frac{\mathrm{Var}[g(X)]}{n}$. Este resultado teórico permítenos obter, mediante a varianza mostral, unha cota para o erro da estimación.

En resumo, os métodos Monte Carlo permítennos facer estimacións cun erro controlado aproveitando toda a capacidade de computación da que dispoñamos. O único requisito é dispoñer dun método rápido de mostraxe sobre a distribución apropiada, problema que trataremos na seguinte sección.

Métodos Monte Carlo en cadeas de Markov (MCMC)

Na Sección 2.1.3 introduciuse o problema que supón simular mostras nun modelo gráfico non dirixido. Se dúas variables X e Y están relacionadas, ambas dependen da outra, e polo tanto non podemos escoller unha orde para simulalas de xeito condicional, como facíamos na introdución.

Nestes casos, empregaremos os chamados métodos Monte Carlo en cadeas de Markov (MCMC polas siglas en inglés). A idea intuitiva é comezar cun vector (x_0, y_0) inicializado de forma aleatoria e irlle facendo actualizacións iterativamente de xeito que, a partir dun certo n, os vectores $(x_n, y_n), (x_{n+1}, y_{n+1}), ...$ sigan a distribución distribución (X, Y) que tiñamos dificultades para simular.

De forma máis xeral, unha cadea de Markov describe unha sucesión de estados onde cada estado depende exclusivamente do estado anterior, carecendo polo tanto de "memoria" respecto ao resto de estados anteriores. Esta "ausencia de memoria" é o que se coñece como *propiedade de Markov*. Formalmente, sexa X unha variable aleatoria con unha densidade descoñecida e soporte $\mathcal{S}$ (discreto ou continuo). A cadea defínese mediante un estado inicial x_0 e unha distribución de transición $T(X_{n+1} \mid X_n)$, que é independente do valor de n e proporciona a distribución do seguinte estado da cadea, dado o estado actual. Se $\mathcal{S}$ é discreto, serán as probabilidades

$$p_{x \to y} \coloneqq \mathbb{P}(X_{n+1} = y \mid X_n = x) = \cdots = \mathbb{P}(X_1 = y \mid X_0 = x),$$

mentres que se X é continua, será unha función de densidade $t_x(X_{n+1})$ que determina a densidade de X_{n+1} unha vez coñecido o valor de $X_n = x$. Fixémonos que a propiedade de Markov non implica que X_{n+1} sexa independente de X_0, senón que, unha vez coñecido X_n, si é independente das observacións anteriores:

$$\mathbb{P}\left(X_{n+1} = y \mid X_n = x_n, \dots, X_0 = x_0\right) = \mathbb{P}\left(X_{n+1} = y \mid X_n = x_n\right).$$

Tendo isto en conta, podemos deducir a expresión correspondente a "saltar" n pasos da cadea:

$$\begin{aligned} p^{(n)}_{x_0\to y} \coloneqq \mathbb{P}(X_n = y \mid X_0 = x_0) &= \sum_{x\in\mathcal{S}} \mathbb{P}(X_n = y, X_{n-1} = x \mid X_0 = x_0) \\ &= \sum_{x\in\mathcal{S}} \mathbb{P}\left(X_n = y \mid X_{n-1} = x, X_0 = x_0\right) \cdot \mathbb{P}\left(X_{n-1} = x \mid X_0 = x_0\right) \\ \text{(prop. Markov)} &= \sum_{x\in\mathcal{S}} \mathbb{P}\left(X_n = y \mid X_{n-1} = x\right) \cdot \mathbb{P}\left(X_{n-1} = x \mid X_0 = x_0\right) \\ &= \sum_{x\in\mathcal{S}} p^{(1)}_{x\to y} \cdot p^{(n-1)}_{x_0\to x}. \end{aligned}$$

Obsérvese que, pese a ser natural descompoñer o estado n utilizando o anterior, $n-1$, a dedución é igualmente válida para calquera estado da cadea m tal que $0 \leq m \leq n$. Esta versión coñécese como a ecuación de Chapman-Kolmogorov:

$$p^{(n)}_{x\to y} = \sum_{z\in\mathcal{S}} p^{(n-m)}_{z\to y} \cdot p^{(m)}_{x\to z} \tag{2.5}$$

e admite unha formulación análoga para estados correspondentes a variables aleatorias continuas

$$t^{(n)}_x(y) = \int_{\mathcal{S}} t^{(n-m)}_z(y) \cdot t^{(m)}_x(z)dz.$$

Recapitulando, estamos buscando unha cadea de Markov $\{X_1, X_2, ... X_n, ...\}$ cuxa distribución aproxime no límite unha distribución obxectivo, de xeito que as mostras xeradas pola cadea sigan dita distribución. Para que exista tal límite, é unha condición necesaria que, chegados a un certo n, os estados da cadea $X_{n+1}, X_{n+2}, ...$ sigan todos a a mesma distribución. É dicir,

$$X_n \sim \pi \quad \overset{T}{\Rightarrow} \quad X_{n+1} \sim \pi \quad \overset{T}{\Rightarrow} \quad X_{n+2} \sim \pi \quad \overset{T}{\Rightarrow} \quad \cdots . \tag{2.6}$$

Unha distribución π sobre o espazo de probabilidade dos estados que satisfaga tal requisito (é dicir, $\mathbb{P}(X_i = x) = \pi(x), \forall i > n$) denomínase *distribución estacionaria* sobre a cadea. Nótese que isto non quere dicir que as distribucións $T(Y \mid X = x)$ non cambien con x. Caractericemos estas distribucións mediante a chamada *ecuación estacionaria*:

$$\begin{aligned} \pi(y) = \mathbb{P}(X_{n+1} = y) &= \sum_{x\in\mathcal{S}} \mathbb{P}(X_{n+1} = y, X_n = x) \\ &= \sum_{x\in\mathcal{S}} \mathbb{P}(X_{n+1} = y \mid X_n = x) \cdot \mathbb{P}(X_n = x) \\ &= \sum_{x\in\mathcal{S}} \pi(x) \cdot p_{x\to y}. \end{aligned}$$

Debido á propiedade transitiva das implicacións na ecuación (2.6), temos que $X_n \sim \pi \Rightarrow X_{n+m} \sim \pi$, do que se deduce a forma xeral da forma estacionaria:

$$\pi(y) = \sum_{x\in\mathcal{S}} \pi(x) \cdot p^{(n)}_{x\to y}, \quad \text{ou} \quad \pi(y) = \int_{\mathcal{S}} \pi(x) \cdot t^{(n)}_x(y)dx, \tag{2.7}$$

na súa forma continua.

Chegados a este punto, xa temos todos os ingredientes para culminar a xustificación teórica do emprego de cadeas de Markov para mostraxe nos métodos MCMC. Utilizando a ecuación estacionaria e a de Chapman-Kolmogorov, probaremos que, se os límites

$$\lim_{n\to\infty} p_{x_0\to y}^{(n)} \quad \text{ou} \quad \lim_{n\to\infty} t_{x_0}^{(n)}(y)$$

existen e son independentes de x_0, entón eses límites definen unha distribución de probabilidade que é estacionaria. Este resultado proba que baixo certas condicións, a mostraxe repetida dunha cadea de Markov inicializada aleatoriamente acabará por devolver mostras que seguen a distribución estacionaria.

Para probar o caso discreto (o continuo é análogo), denotemos $\pi(y) = \lim_{n\to\infty} p_{x_0\to y}^{(n)}$ e comecemos observando que efectivamente é unha masa de probabilidade:

$$\sum_{y\in\mathcal{S}} \pi(y) = \sum_{y\in\mathcal{S}} \lim_{n\to\infty} p_{x_0\to y}^{(n)} = \lim_{n\to\infty} \sum_{y\in\mathcal{S}} p_{x_0\to y}^{(n)} = \lim_{n\to\infty} 1 = 1.$$

Agora, comprobemos que cumpre a ecuación estacionaria (2.7), empregando a ecuación de Chapman-Kolmogorov (2.5):

$$\sum_{x\in\mathcal{S}} \pi(x)p_{x\to y} = \sum_{x\in\mathcal{S}} \left(\lim_{n\to\infty} p_{x_0\to x}^{(n)}\right) p_{x\to y} = \lim_{n\to\infty} \sum_{x\in\mathcal{S}} p_{x_0\to x}^{(n)} \cdot p_{x\to y} \overset{C.K}{=} \lim_{n\to\infty} p_{x_0\to y}^{(n+1)} = \pi(y).$$

Esta sección condensou as definicións e propiedades básicas das cadeas de Markov no contexto dos métodos MCMC. Ross (1996) e Kulkarni (2016) desenvolven as súas propiedades con máis detalle. Cando se apliquen estes métodos, o estudo do problema concreto e da distribución de transición deberá preocuparse da existencia da distribución límite, das propiedades de converxencia da cadea e do valor inicial x_0.

Rematemos sinalando os dous problemas principais dos que adoecen os métodos MCMC. Por unha parte, o marco teórico proporciona garantías de que, existindo a distribución límite, a cadea converxerá nalgún momento, pero non facilita unha cota para as iteracións necesarias. O proceso de realizar iteracións ata alcanzar a distribución límite coñécense como *queimado da cadea* e, na práctica, empréganse heurísticas e outras técnicas pouco satisfactorias para resolver este problema.

Por outra parte, se ben as mostras que xeradas na distribución límite son identicamente distribuídas, non son independentes. Pola construción das cadeas, hai unha forte correlación entre mostras sucesivas. Este fenómeno, denominado *mestura pobre*, é especialmente problemático en distribucións multimodais cunha gran separación entre modas (Figura 2.3), pois a cadea tenderá

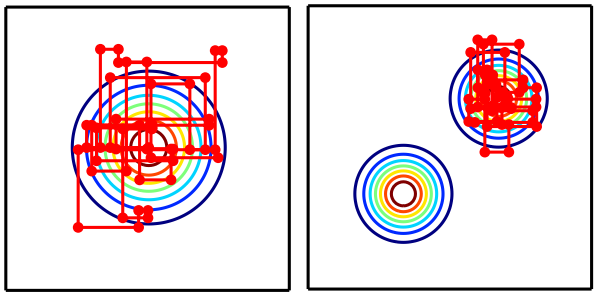


Figura 2.3: Exemplos de MCMC reproducidos de Goodfellow et al. (2016). Na esquerda, un comportamento óptimo dun método MCMC. Apréciase que a distribución global é correcta e que as mostras consecutivas están correlacionadas. Na dereita, o fenómeno de mestura pobre nunha distribución bimodal.

a ir xerando valores ao redor dunha das modas e seralle moi complicado "escapar" se non hai unha rexión con densidade suficiente entre as modas.

Para solucionar este problema, pódense simular varias cadeas de Markov en paralelo, co que as mostras obtidas serán verdadeiramente independentes. Outra alternativa menos custosa computacionalmente para diminuír a correlación é descartar varios estados entre mostras, de xeito que non sexan consecutivas. Na práctica, chegarase a un compromiso entre o número de cadeas e as veces que se simula en cada cadea.

Existen diversos algoritmos MCMC con diferentes propiedades, garantías de converxencia e complexidade nas implementacións. No caso de modelos gráficos, a referencia indiscutible é o método de Gibbs, que estudaremos a continuación.

Método de Gibbs

O método de Gibbs explota a estrutura especial de certos modelos para construír un método MCMC que simula mostras moi eficientemente. En xeral, é válido para modelos multivariantes nos que sexa sinxelo obter mostras de certas distribucións condicionais, pero sexa intratable simular as marxinais. Como vimos na Sección 2.1.3, é xustamente o caso dos modelos gráficos non dirixidos.

No contexto de modelos gráficos, o método de Gibbs formúlase habitualmente para modelos baseados en enerxía ou máquinas de Boltzmann, introducidos na Sección 2.1.1. As máquinas de Boltzmann presentan unha vantaxe fundamental: o potencial de cada clan de nodos é sempre estritamente positivo, grazas á acción da función exponencial. Koller e Friedman (2009) desenvolven os resultados que proban a converxencia do método de Gibbs á distribución conxunta para calquera estado inicial en todo campo aleatorio de Markov con potenciais dos clans positivos.

Describamos o funcionamento do método de Gibbs de xeito esquemático. Consideremos un modelo gráfico con grafo $\mathcal{G}$ e vector de variables aleatorias asociado $\boldsymbol{X} = (X_1, \dots X_n)$. Este vector define unha ordenación arbitraria das variables. Sexa $\boldsymbol{x}^{(0)} = (x_1^{(0)}, \dots, x_n^{(0)})$ un vector correspondente ao estado inicial, onde o superíndice denota a iteración. En cada iteración, Gibbs modifica todas as variables por orde, obtendo unha mostra de cada variable condicionada no resto. É dicir, na primeira iteración, comezará obtendo $x_1^{(1)} \sim \mathbb{P}(X_1 \mid X_2 = x_2^{(0)}, \dots, X_n = x_n^{(0)})$, despois $x_2^{(1)} \sim \mathbb{P}(X_2 \mid X_1 = x_1^{(1)}, X_3 = x_3^{(0)}, \dots, X_n = x_n^{(0)})$ e así sucesivamente. O algoritmo 1 amosa formalmente o método.

Algoritmo 1 Método de mostraxe de Gibbs

$\boldsymbol{x}^{(0)} \leftarrow (x_1^{(0)}, \dots x_n^{(0)})$
for $t = 1, \dots, T_{\text{máx}}$ **do**
  for $i = 1, \dots, n$ **do**
    $\boldsymbol{x}^{(t)} \leftarrow \boldsymbol{x}^{(t-1)}$
    $x_i^{(t)} \sim \mathbb{P}(X_i \mid \{X_j = x_j^{(t)}\}_{j=1\dots n, j\neq i})$
  end for
end for
return $\boldsymbol{x}^{(I)}, \boldsymbol{x}^{(I+1)}, \dots, \boldsymbol{x}^{(T)}$

O único requisito do método de Gibbs é que sexa sinxelo obter mostras en cada variable condicionando nas demais. Débese ter en conta que as primeiras $(I-1)$ iteracións producen mostras que non están ben distribuídas, ata que se complete o queimado da cadea e a distribución estea próxima á estacionaria. En función do modelo, estableceranse diferentes heurísticas para identificar este punto I.

O método de Gibbs é un caso particular dun algoritmo máis xeral, o de Metropolis-Hastings, que é aplicable a un conxunto máis amplo de modelos.

2.1.5. Optimización de parámetros en modelos non dirixidos

Supoñamos que temos un campo aleatorio de Markov paramétrico que modela o vector aleatorio $\boldsymbol{X} = (X_1, \dots, X_n)$. A súa función de densidade será

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{Z_{\boldsymbol{\theta}}} \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{x}), \text{ sendo } Z_{\boldsymbol{\theta}} = \int_{\mathcal{S}} \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{u}) d\boldsymbol{u}.$$

onde $\boldsymbol{\theta}$ é o vector de parámetros do modelo.

Para axustar $\boldsymbol{\theta}$ en base a unha mostra $\boldsymbol{X}^{(1)}, \dots, \boldsymbol{X}^{(M)}$ seguindo técnicas de máxima verosimilitude, debemos resolver o problema de optimización presentado na ecuación (1.2) da Sección

1.2.1. Por suposto, nesta clase de modelos non se pode calcular a solución exacta do problema, senón que se debe realizar unha optimización numérica de $\boldsymbol{\theta}$. A aproximación habitual é empregar un método de descenso do gradiente, que actualizará $\boldsymbol{\theta}$ iterativamente a través do gradiente $\nabla_{\boldsymbol{\theta}} \log f_{\boldsymbol{\theta}}(\boldsymbol{x})$:

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \lambda \cdot \nabla_{\boldsymbol{\theta}} \left[\sum_{i=1}^{M} \log f_{\boldsymbol{\theta}} \left(\boldsymbol{X}^{(i)} \right) \right] = \boldsymbol{\theta}_i - \lambda \sum_{i=1}^{M} \left[\nabla_{\boldsymbol{\theta}} \log f_{\boldsymbol{\theta}} \left(\boldsymbol{X}^{(i)} \right) \right].$$

O gradiente necesario pódese expresar como

$$\nabla_{\boldsymbol{\theta}} \log f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \nabla_{\boldsymbol{\theta}} \log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{x}) - \nabla_{\boldsymbol{\theta}} \log Z_{\boldsymbol{\theta}}. \tag{2.8}$$

Na maioría de modelos, o termo da esquerda é fácil de computar. O termo da dereita, pola contra, require do cálculo da integral intratable de $Z_{\boldsymbol{\theta}}$. A continuación, vexamos como aproximar este termo mediante métodos Monte Carlo:

$$\nabla_{\boldsymbol{\theta}} \log Z_{\boldsymbol{\theta}} = \frac{\nabla_{\boldsymbol{\theta}} Z_{\boldsymbol{\theta}}}{Z_{\boldsymbol{\theta}}} = \frac{\nabla_{\boldsymbol{\theta}} \int \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{u}) d\boldsymbol{u}}{Z_{\boldsymbol{\theta}}} = \frac{\int \nabla_{\boldsymbol{\theta}} \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{u}) d\boldsymbol{u}}{Z_{\boldsymbol{\theta}}}.$$

A aplicación da regra de Leibniz na última igualdade para o cálculo diferencial baixo a integral impón dúas condicións sobre $\tilde{f}_{\boldsymbol{\theta}}$. Debe ser integrable no sentido de Lebesgue sobre $\boldsymbol{x}$ para todo $\boldsymbol{\theta}$ e $\nabla_{\boldsymbol{\theta}} \tilde{f}_{\boldsymbol{\theta}}$ debe existir e estar limitado para todo $\boldsymbol{\theta}$ e para case todo $\boldsymbol{x}$. No contexto dos modelos non dirixidos habituais, estas asuncións non adoitan ser un problema. Se traballamos con variables discretas, a integral é unha suma sobre o soporte e as condicións desaparecen por completo.

En modelos que garantan $\tilde{f}_{\boldsymbol{\theta}} > 0$ (como as máquinas de Boltzmann da Sección 2.1.1), podemos substituír $\tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{x})$ por $e^{\log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{x})}$:

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} \log Z_{\boldsymbol{\theta}} = \frac{\int \nabla_{\boldsymbol{\theta}} e^{\log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{u})} d\boldsymbol{u}}{Z_{\boldsymbol{\theta}}} &= \frac{\int e^{\log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{u})} \nabla_{\boldsymbol{\theta}} \log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{u}) d\boldsymbol{u}}{Z_{\boldsymbol{\theta}}} && (2.9)\\
&= \frac{\int \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{u}) \nabla_{\boldsymbol{\theta}} \log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{u}) d\boldsymbol{u}}{Z_{\boldsymbol{\theta}}} && (2.10)\\
&= \int_{\mathcal{S}} f(\boldsymbol{u}) \nabla_{\boldsymbol{\theta}} \log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{u}) d\boldsymbol{u} && (2.11)\\
&= \mathbb{E}_{\boldsymbol{X}} \left[\nabla_{\boldsymbol{\theta}} \log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{X}) \right]. && (2.12)
\end{aligned}$$

A ecuación (2.12) permítenos construír un estimador Monte Carlo para $\nabla_{\boldsymbol{\theta}} \log Z_{\boldsymbol{\theta}}$, que denotaremos como $\hat{s}$, sen máis que simular unha mostra $\hat{\boldsymbol{X}}^{(1)}, \ldots, \hat{\boldsymbol{X}}^{(M')}$ a partir do modelo:

$$\hat{s} = \frac{1}{M} \sum_{i=1}^{M'} \nabla_{\boldsymbol{\theta}} \log \tilde{f}_{\boldsymbol{\theta}}(\hat{\boldsymbol{X}}^{(i)}).$$

A mostra pódese simular empregando o método de Gibbs visto na sección anterior. Deste xeito, o cálculo do gradiente para actualizar $\boldsymbol{\theta}$ descrito na ecuación (2.8) realizaríase en dous

pasos. Primeiro, calcularíase unha parte do gradiente, de signo positivo, que asigna máis probabilidade a un dato da mostra real da que dispoñemos. En segundo lugar, calcúlase $\hat{s}$, que non depende da mostra real e con signo negativo na ecuación (2.8), de xeito que se asigne menos probabilidade aos datos xerados polo modelo.

Intuitivamente, obsérvase como este procedemento converxerá cando as distribucións do modelo e dos datos sexan próximas, pois as actualizacións que realizan ambas fases sobre $\boldsymbol{\theta}$ tenderán a cancelarse asintoticamente.

Na práctica, algoritmos coñecidos coma o de *diverxencia contrastiva* (CD) ou o de *máxima verosimilitude estocástica* (SML) baséanse neste procedemento, diferindo soamente en certas variacións que introducen para limitar o custo computacional da simulación de mostras da fase negativa.

2.1.6. Unha introdución á inferencia variacional e de campo medio

Os métodos Monte Carlo permiten aproximar as integrais ou sumatorios intratables que presentan os problemas de inferencia en modelos non dirixidos. Posibilitan algoritmos como o *Importance Sampling*, que proporcionan un marco xeral para estimar a constante de normalización Z ou as integrais das distribucións marxinais. Neste traballo, non o desenvolveremos por non ser relevante para os modelos presentados, pero pode consultarse en Goodfellow et al. (2016). A sección anterior amosou unha aplicación particular dos métodos Monte Carlo na implementación de regras de axuste baseadas en técnicas de máxima verosimilitude.

Unha aproximación alternativa á inferencia en modelos non dirixidos, que terá moita importancia na formulación de NADE no Capítulo 3, son as formulacións de métodos de inferencia como un problema de optimización. Intuitivamente, estes métodos consisten en minimizar a diverxencia de Kullback-Leibler (Sección 1.2.2) entre a distribución do modelo e unha distribución arbitraria sobre a cal impoñemos unha restrición non paramétrica. Como a diverxencia KL é non negativa, e 0 só se as distribucións son iguais, é unha boa medida de como de próximas están.

A beleza desta aproximación reside en que, minimizando analiticamente a diverxencia e empregando ferramentas como o cálculo de variacións (de aí o nome de inferencia variacional), pódense achar expresións exactas para función de densidade da nova distribución. A restrición non paramétrica encargarase de que a expresión de dita función de densidade sexa facilmente computable.

A restrición non paramétrica máis frecuente (véxase Koller e Friedman (2009)) para estes casos é a chamada aproximación de campo medio, un concepto inspirado pola física e introducido

na teoría de modelos gráficos por Saul e Jordan (1995). Consiste en asumir que unha función de densidade dun vector multidimensional factoriza nun produto de densidades unidimensionais, é dicir:

$$f(\boldsymbol{x}) = \prod_{i=1}^{n} f(x_i).$$

Na Sección 3.1 veremos unha aplicación práctica desta aproximación.

2.1.7. A máquina de Boltzmann restrinxida

A máquina de Boltzmann restrinxida (RBM), presentada en Smolensky (1986), é o exemplo típico de modelo non dirixido. É un modelo baseado en enerxía (Sección 2.1.1) coa particularidade de que o grafo ten unha estrutura bipartita: sepárase en dous bloques que corresponden ás variables visibles ($\boldsymbol{V}$) e latentes ($\boldsymbol{H}$), sen conexións entre variables do mesmo bloque. Esta condición bipartita é a que lle da o nome de "restrinxida", mentres que os bloques conéctanse de forma totalmente densa entre si, como apreciamos na Figura 2.4.

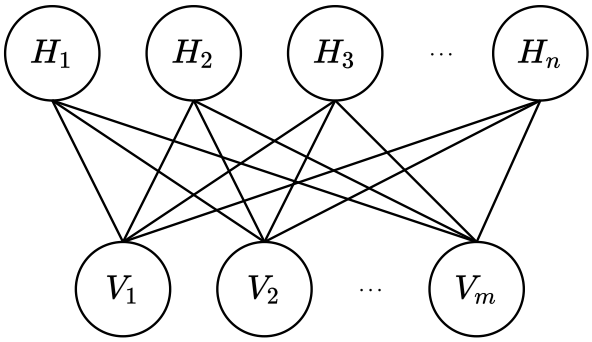


Figura 2.4: Grafo dunha máquina de Bolzmann restrinxida.

A cantidade de variables latentes e a liberdade paramétrica que teñen para "conectar" con todas as visibles dotan ao modelo dunha gran capacidade para modelar relacións complexas entre as variables visibles. A función de enerxía da máquina de Boltzmann restrinxida é

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\boldsymbol{b}^\top \boldsymbol{v} - \boldsymbol{c}^\top \boldsymbol{h} - \boldsymbol{h}^\top \boldsymbol{W} \boldsymbol{v},$$

onde $\boldsymbol{b} \in \mathbb{R}^m$, $\boldsymbol{c} \in \mathbb{R}^n$ e $\boldsymbol{W} \in \mathcal{M}_{n\times m}(\mathbb{R})$ son parámetros sen restricións e $\boldsymbol{v} \in \mathbb{R}^m$, $\boldsymbol{h} \in \mathbb{R}^n$, de xeito que toda interacción entre $\boldsymbol{V}$ e $\boldsymbol{H}$ ven dada pola matriz $\boldsymbol{W}$. Esta estrutura restrinxida é responsable de varias propiedades moi desexables para realizar inferencia e axustar o modelo. Por exemplo, a independencia condicional entre os elementos de $\boldsymbol{V}$ ou de $\boldsymbol{H}$ dado o outro grupo

permite descompoñer o cálculo das probabilidades condicionais nun produto:

$$\mathbb{P}\left(\boldsymbol{H}=\boldsymbol{h} \mid \boldsymbol{v}\right)=f(\boldsymbol{h} \mid \boldsymbol{v})=\prod_{i=1}^{n} f\left(h_i \mid \boldsymbol{v}\right), \tag{2.13}$$

$$\mathbb{P}\left(\boldsymbol{V}=\boldsymbol{v} \mid \boldsymbol{h}\right)=f(\boldsymbol{v} \mid \boldsymbol{h})=\prod_{j=1}^{m} f\left(v_j \mid \boldsymbol{h}\right). \tag{2.14}$$

Vexamos unha aplicación sinxela da RBM empregando a súa formulación orixinal con variables binarias. Neste caso, a estrutura especial da RBM facilita o cálculo das probabilidades individuais que conforman a función masa de probabilidade:

$$f(h_i \mid \boldsymbol{v})=\mathbb{P}\left(H_i=1 \mid \boldsymbol{v}\right)^{h_i}\left(1-\mathbb{P}\left(H_i=1 \mid \boldsymbol{v}\right)\right)^{1-h_i}, \tag{2.15}$$

$$\mathbb{P}\left(H_i=1 \mid \boldsymbol{v}\right)=\sigma\left(\boldsymbol{W}_{i,\cdot}\boldsymbol{v}+c_i\right)=\sigma\left(\sum_{j=1}^{m} W_{ij}v_j+c_i\right), \tag{2.16}$$

$$f(v_j \mid \boldsymbol{h})=\mathbb{P}\left(V_j=1 \mid \boldsymbol{h}\right)^{v_j}\left(1-\mathbb{P}\left(V_j=1 \mid \boldsymbol{h}\right)\right)^{1-v_j}, \tag{2.17}$$

$$\mathbb{P}\left(V_j=1 \mid \boldsymbol{h}\right)=\sigma\left(\boldsymbol{h}^{\top}\boldsymbol{W}_{\cdot,j}+b_j\right)=\sigma\left(\sum_{i=1}^{n} W_{ij}h_i+b_j\right), \tag{2.18}$$

onde, novamente, $\sigma(x)=\frac{1}{1+e^{-x}}$ é a función sigmoide. Estas dúas igualdades son sorprendentemente sinxelas, e a súa expresión coincide coa formulación matricial dunha rede neuronal (vista na Sección 1.3) dunha capa con m ou n entradas e unha única neurona na capa de saída.

Probemos a igualdade (2.18), sendo o razoamento análogo para a ecuación (2.16). Denotemos como $\boldsymbol{v}_{-l}$ o conxunto das observacións de todas as variables visibles salvo a de índice l. Definamos

$$\alpha_l(\boldsymbol{h}):=-b_l-\sum_{i=1}^{n} w_{il}h_i, \quad \beta\left(\boldsymbol{v}_{-l},\boldsymbol{h}\right):=-\sum_{i=1}^{n}\sum_{j=1,j\neq l}^{m} w_{ij}h_iv_j-\sum_{j=1,j\neq l}^{m} b_iv_i-\sum_{i=1}^{n} c_ih_i,$$

de xeito que α_l extrae os termos de E que teñan a v_l, e $E(\boldsymbol{v},\boldsymbol{h})=\beta\left(\boldsymbol{v}_{-l},\boldsymbol{h}\right)+v_l\alpha_l(\boldsymbol{h})$. Agora:

$$\begin{aligned}
\mathbb{P}\left(V_l=1 \mid \boldsymbol{h}\right)&=\mathbb{P}\left(V_l=1 \mid \boldsymbol{v}_{-l},\boldsymbol{h}\right)=\frac{\mathbb{P}\left(V_l=1,\boldsymbol{V}_{-l}=\boldsymbol{v}_{-l},\boldsymbol{H}=\boldsymbol{h}\right)}{\mathbb{P}\left(\boldsymbol{V}_{-l}=\boldsymbol{v}_{-l},\boldsymbol{H}=\boldsymbol{h}\right)}\\
&=\frac{e^{-E(v_l=1,\boldsymbol{v}_{-l},\boldsymbol{h})}}{e^{-E(v_l=1,\boldsymbol{v}_{-l},\boldsymbol{h})}+e^{-E(v_l=0,\boldsymbol{v}_{-l},\boldsymbol{h})}}=\frac{e^{-\beta(\boldsymbol{v}_{-l},\boldsymbol{h})-1\alpha_l(\boldsymbol{h})}}{e^{-\beta(\boldsymbol{v}_{-l},\boldsymbol{h})-1\alpha_l(\boldsymbol{h})}+e^{-\beta(\boldsymbol{v}_{-l},\boldsymbol{h})-0\alpha_l(\boldsymbol{h})}}\\
&=\frac{e^{-\beta(\boldsymbol{v}_{-l},\boldsymbol{h})}e^{-\alpha_l(\boldsymbol{h})}}{e^{-\beta(\boldsymbol{v}_{-l},\boldsymbol{h})}e^{-\alpha_l(\boldsymbol{h})}+e^{-\beta(\boldsymbol{v}_{-l},\boldsymbol{h})}}=\frac{e^{-\beta(\boldsymbol{v}_{-l},\boldsymbol{h})}e^{-\alpha_l(\boldsymbol{h})}}{e^{-\beta(\boldsymbol{v}_{-l},\boldsymbol{h})}\left(e^{-\alpha_l(\boldsymbol{h})}+1\right)}\\
&=\frac{e^{-\alpha_l(\boldsymbol{h})}}{e^{-\alpha_l(\boldsymbol{h})}+1}=\frac{\frac{1}{e^{\alpha_l(\boldsymbol{h})}}}{\frac{1}{e^{\alpha_l(\boldsymbol{h})}}+1}=\frac{1}{1+e^{\alpha_l(\boldsymbol{h})}}\\
&=\sigma\left(-\alpha_l(\boldsymbol{h})\right)=\sigma\left(\sum_{i=1}^{n} w_{il}h_i+b_l\right).
\end{aligned}$$

As propiedades das ecuacións (2.13)-(2.14)-(2.16)-(2.18) permiten que a obtención de mostras da RBM sexa moi sinxela mediante o método de Gibbs. Ademais, nin sequera hai que iterar todas as variables, sendo posible simular paralelamente os bloques das $\boldsymbol{V}$ e das $\boldsymbol{H}$.

Ilustremos, mediante a RBM con variables binarias, o termo "integral intratable" que leva aparecendo en todo o Capítulo. Cabe destacar que o feito de que as variables sexan discretas e binarias xoga moi a favor da facilidade de computación, polo que este exemplo servirá para imaxinar a magnitude do problema en canto non se den estas condicións. Consideremos, por exemplo, unha RBM con 100 variables visibles e 100 latentes, e poñamos que queremos calcular a constante de normalización Z:

$$Z = \sum_{\boldsymbol{v},\boldsymbol{h}} \hat{f}(\boldsymbol{v},\boldsymbol{h}) = \sum_{v_1=0}^{1}\sum_{v_2=0}^{1}\cdots\sum_{v_{100}=0}^{1}\sum_{h_1=0}^{1}\sum_{h_2=0}^{1}\cdots\sum_{h_{100}=0}^{1} e^{-E(\boldsymbol{v},\boldsymbol{h})}.$$

Cantas veces habería que avaliar $e^{-E(\boldsymbol{v},\boldsymbol{h})}$? Como vemos, trátase dunha ramificación exponencial onde en cada sumatorio hai dous posibles valores, polo que habería que avaliar o termo $2^{100} = 1,26 \cdot 10^{30}$ veces! O supercomputador máis potente do mundo, o estadounidense *Frontier* con 10^{18} operacións de coma flotante por segundo, tardaría 32.000 anos en computar a Z deste modelo tan sinxelo, asumindo por simplicidade que a avaliación constase só dunha operación.

Axuste dunha RBM

O axuste das RBM pode realizarse con varias aproximacións, que se resumen en Fischer e Igel (2012). Os métodos máis habituais son os procedementos baseados na estimación do gradiente respecto aos parámetros, tal e como presentamos na Sección 2.1.5. Segundo o exposto, todo o que precisamos é computar $\nabla_{\boldsymbol{\theta}} \log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{v})$, onde $\boldsymbol{\theta}$ é o vector de parámetros da RBM:

$$\boldsymbol{\theta} = (b_1, \dots, b_m, c_1, \dots, c_n, W_{1,1}, \dots, W_{n,1}, W_{1,2}, \dots, W_{n,2}, \dots, W_{1,m}, \dots, W_{n,m}).$$

Para iso, debemos calcular a distribución marxinal respecto as variables latentes:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} \log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{v}) &= \frac{\partial}{\partial \boldsymbol{\theta}}\left(\log \sum_{\boldsymbol{h}} e^{-E_{\boldsymbol{\theta}}\boldsymbol{v},\boldsymbol{h})}\right) = -\frac{1}{\sum_{\boldsymbol{h}} e^{-E_{\boldsymbol{\theta}}\boldsymbol{v},\boldsymbol{h})}} \sum_{\boldsymbol{h}} e^{-E_{\boldsymbol{\theta}}\boldsymbol{v},\boldsymbol{h})} \cdot \frac{\partial E_{\boldsymbol{\theta}}\boldsymbol{v},\boldsymbol{h})}{\partial \boldsymbol{\theta}} \\ &= -\sum_{\boldsymbol{h}} f(\boldsymbol{h} \mid \boldsymbol{v}) \frac{\partial E_{\boldsymbol{\theta}}\boldsymbol{v},\boldsymbol{h})}{\partial \boldsymbol{\theta}}.\end{aligned}$$

Recordemos que $E_{\boldsymbol{\theta}}(\boldsymbol{v},\boldsymbol{h}) = -\boldsymbol{b}^\top\boldsymbol{v} - \boldsymbol{c}^\top\boldsymbol{h} - \boldsymbol{h}^\top\boldsymbol{W}\boldsymbol{v}$. Comecemos calculando o gradiente respecto aos parámetros da matriz $\boldsymbol{W}$. Temos que $\frac{\partial E_{\boldsymbol{\theta}}\boldsymbol{v},\boldsymbol{h})}{\partial W_{i,j}} = -v_i h_j$, de xeito que o gradiente respecto a $W_{i,j}$ sería:

$$\begin{aligned}\nabla_{W_{i,j}} \log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{v}) &= \sum_{\boldsymbol{h}} f(\boldsymbol{h} \mid \boldsymbol{v}) v_i h_j = \sum_{\boldsymbol{h}} \prod_{k=1}^{n} f\left(h_k \mid \boldsymbol{v}\right) h_i v_j \\ &= \sum_{h_i=0}^{1} \sum_{\boldsymbol{h}_{-i}} f\left(h_i \mid \boldsymbol{v}\right) f\left(\boldsymbol{h}_{-i} \mid \boldsymbol{v}\right) h_i v_j = \sum_{h_i=0}^{1} f\left(h_i \mid \boldsymbol{v}\right) h_i v_j \sum_{\boldsymbol{h}_{-i}} f\left(\boldsymbol{h}_{-i} \mid \boldsymbol{v}\right)\end{aligned}$$

$$= \mathbb{P}\left(H_i = 1 \mid \boldsymbol{v}\right) v_j = \sigma\left(\sum_{j=1}^{m} w_{ij} v_j + c_i\right) v_j.$$

Para os parámetros b_j obtense de xeito análogo $\nabla_{b_j} \log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{v}) = v_j$ e para os c_i, $\nabla_{c_i} \log \tilde{f}_{\boldsymbol{\theta}}(\boldsymbol{v}) = \mathbb{P}\left(H_i = 1 \mid \boldsymbol{v}\right)$.

Implementación real

Coas ecuacións anteriores xa temos todo o necesario para implementar unha RBM e axustala sobre o conxunto de datos do MNIST binario. Empregaremos $28 \cdot 28 = 784$ variables visibles e 500 variables latentes. Adestraremos o modelo facendo 15 pasadas polo conxunto de datos MNIST empregando o algoritmo da Sección 2.1.5 con $\lambda = 0{,}1$. A Figura 2.4 amosa mostras xeradas despois do adestramento, empregando 20 cadeas de Markov.

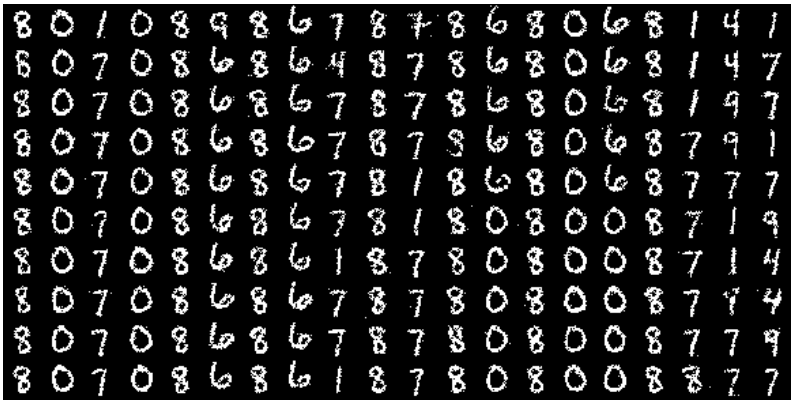


Figura 2.5: Mostras simuladas por unha RBM mediante o método de Gibbs. Cada columna correponde a unha cadea de Markov distinta, nas que podemos observar o fenómeno da mestura pobre.

Ademais, a implementación permite visualizar facilmente a matriz de pesos $\boldsymbol{W}$. Se nos fixamos na ecuación (2.16), a fila i-ésima $\boldsymbol{W}_{i,\cdot}$ contén os pesos para calcular a probabilidade de activación da variable latente H_i, dada unha imaxe $\boldsymbol{v}$. Así, podemos representar cada fila da matriz como unha imaxe 28×28. A probabilidade de activación da variable latente correspondente será maior canto máis se asemelle a imaxe de entrada $\boldsymbol{v}$ coas zonas claras na representación dos pesos. A continuación, a Figura 2.6 amosa as representacións das primeiras 100 filas de $\boldsymbol{W}$.

En resumo, as RBM proporcionan un modelo non dirixido onde o cómputo das probabilidades condicionais máis importantes é inmediato, a simulación de mostras mediante o método de Gibbs é relativamente simple e o axuste do modelo é directo mediante o algoritmo da diverxencia

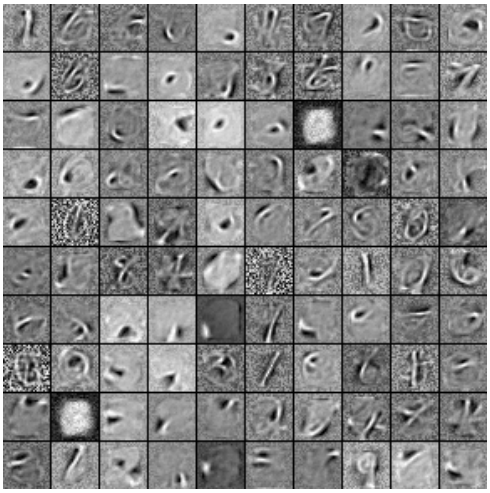


Figura 2.6: Representación gráfica das primeiras 100 filas da matriz $\boldsymbol{W}$. Cada imaxe correspóndese cunha fila de $\boldsymbol{W}$ e cunha variable latente H_i.

contrastiva que acabamos de presentar. Non obstante, a constante de normalización Z segue a ser intratable computacionalmente e a aplicación de métodos Monte Carlo fai que a RBM sexa relativamente lenta respecto a outros modelos.

2.2. Modelos dirixidos ou redes bayesianas

Ao comezo do capítulo, a Figura 2.1, xa presentou un modelo dirixido. Intuitivamente, xa vimos como a estrutura dirixida do grafo facilitaba a factorización da función de densidade e a simulación de mostras.

Antes de profundar nesta clase de modelos, debemos aclarar un abuso de notación típico na literatura. A denominación de modelo dirixido ou rede bayesiana refírese exclusivamente a modelos cuxos grafos son finitos, dirixidos e sen ciclos, pese a non especificarse totalmente no nome. Recordemos que un ciclo nun grafo dirixido é un camiño que comeza e remata no mesmo nodo, sendo un camiño unha secuencia de vértices distintos conectados por arestas de xeito consecutivo.

Grazas a carecer o grafo de ciclos, a función de densidade en cada nodo pode expresarse en función dos nodos pai, sen que aparezan bucles. Denotemos como $\mathcal{P}(X_i)$ os pais no grafo da variable aleatoria X_i. Así, podemos escribir a función de densidade nun modelo dirixido do seguinte modo:

$$f(\boldsymbol{x}) = \prod_{i=1}^{n} f\left(x_i \mid \mathcal{P}(x_i)\right)$$

A Figura 2.7 amosa un exemplo de modelo gráfico dirixido, no cal $\mathcal{P}(X_4) = \{X_1, X_3\}$ e $f(\boldsymbol{x}) = f(x_1)f(x_3)f(x_2 \mid x_1)f(x_4 \mid x_1, x_3)f(x_5 \mid x_2, x_4)$.

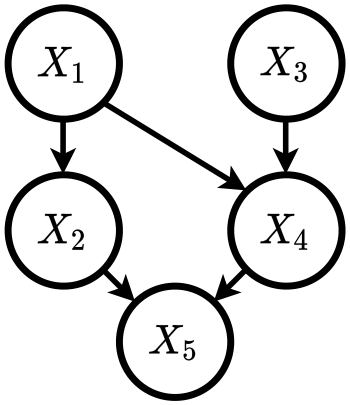


Figura 2.7: Exemplo de modelo gráfico dirixido.

Os grafos dirixidos acíclicos finitos (DAG, polas siglas en inglés) foron amplamente estudados nas Ciencias da Computación pola súa aplicación na construción de compiladores, onde xogan un papel fundamental na optimización e reordenamento das instrucións dun programa. Neste contexto, denomínase orde topolóxica do grafo á orde parcial que confire a relación "pai de" do grafo, de xeito que $X_i > X_j$ se e só se X_i é descendente de X_j. Nótese que non ten porqué estar definida, por exemplo na Figura 2.7 non podemos comparar X_3 con X_2 nin con X_1. Unha propiedade importante é que todo DAG admite polo menos un ordenamento dos nodos segundo a orde topolóxica (véxase Cormen et al. (2022) para un tratamento extenso dos DAG e do algoritmo de obtención da orde topolóxica). No exemplo da figura, dous ordenamentos topolóxicos son $(X_1, X_2, X_3, X_4, X_5)$ e $(X_3, X_1, X_4, X_2, X_5)$. Un ordenamento topolóxico permite recorrer os nodos sen visitar nunca un fillo antes cun pai.

Esta propiedade posibilita un método moi sinxelo de simulación de mostras, o *método ancestral*. O método é tan sinxelo como ir simulando mostras das variables aleatorias individuais seguindo un ordenamento topolóxico do grafo.

En comparación cos modelos non dirixidos, o cómputo da función de densidade nas redes bayesianas é computacionalmente tratable (ao non ter o problema da constante de normalización Z) e dispoñen dun método de mostraxe directo (seguindo a orde topolóxica). Dos problemas que tratamos para os modelos non dirixidos, os único que se manteñen como computacionalmente intratables son, por unha banda, realizar inferencia nos modelos (xa que computar condicionais en modelos dirixidos con variables latentes segue sendo moi custoso) e simular mostras nunha orde que non sexa a topolóxica (neste caso, teremos que empregar de novo métodos como o de Gibbs).

2.2.1. Redes autorregresivas

As redes autorregresivas son modelos dirixidos que non teñen variables latentes e nos que $\mathcal{P}(X_i) = \{X_1, \dots, X_{i-1}\}$. Un exemplo é o modelo LARC que presentamos na Sección 1.4. Todas teñen a mesma estrutura xeral:

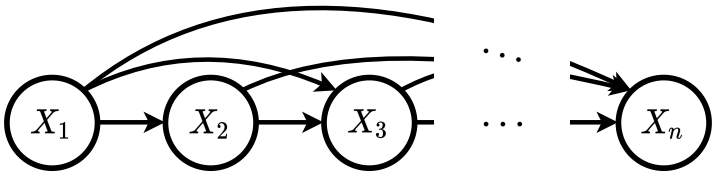


Figura 2.8: Estrutura das redes autorregresivas.

As redes autorregresivas teñen un grafo moi simple, pero adoitan trasladar a complexidade ao modelado das funcións de densidade condicionais $f(x_i \mid x_1, \dots, x_{i-1})$. No caso de LARC (Sección 1.4) estas condicionais modélanse mediante unha simple regresión loxística. No seguinte Capítulo, veremos como NADE recolle a idea e mellora o modelo, mantendo a estrutura da rede autorregresiva, pero empregando redes neuronais para modelar as funcións de densidade condicionais.

En xeral, a redes autorregresivas empréganse con moito éxito combinadas con modelos de regresión para as densidades condicionais. Se os datos son binarios, a regresión será loxística (como LARC) e se son reais, a regresión será linear.

Unha parametrización como a de LARC, que emprega unha función distinta para cada densidade condicional, require de $O(n^2)$ parámetros. Unha posible mellora, que veremos en NADE, é compartir parámetros que se usen no cómputo de $f(x_{i-1} \mid x_1, \dots, x_{i-2})$ para estimar $f(x_i \mid x_1, \dots, x_{i-1})$.

Capítulo 3

NADE: Neural Autoregressive Distribution Estimation

No Capítulo 1 mostramos os problemas que presentan as técnicas clásicas de estimación de densidade en mostras de moi alta dimensión. Isto motiva a introdución de modelos como LARC (Sección 1.4), que perde moita capacidade de modelado para gañar en custo computacional. Posteriormente, o Capítulo 2 introduciu formalmente os modelos gráficos. Presentamos a RBM como exemplo de modelo non dirixido, e vimos como amosa unha enorme capacidade de modelado a costa de requirir un gran aparato técnico e numérico para a súa operación. NADE, o caso de estudo deste traballo, pretende xuntar a capacidade da RBM mantendo a simplicidade das redes autorregresivas, apoiándose en ferramentas como as redes neuronais.

NADE representa as siglas de "*Neural Autoregressive Distribution Estimation*", que podemos traducir como "estimación de densidade autorregresiva neuronal". Trátase dun modelo presentado orixinalmente no 2011 por Hugo Larochelle e Iain Murray, e reeditado en Uria et al. (2016), unha publicación ampliada que recolle tamén desenvolvementos posteriores. Este Capítulo profundará nas bases teóricas e no funcionamento deste modelo.

Trátase dunha rede autorregresiva (Sección 2.2.1), polo que pertence á categoría dos modelos gráficos dirixidos. En esencia, a característica que a distingue doutras redes autorregresivas é a parametrización das funcións de densidade condicionais mediante redes neuronais. A expresión desta parametrización deducirase a partir dunha RBM (Sección 2.1.7).

Recordemos a notación $\boldsymbol{v}_{-l} = (v_1, \dots, v_{l-1}, v_{l+1}, \dots, v_n)$; neste Capítulo será especialmente práctica, e estenderémola con $\boldsymbol{v}_{<l} = (v_1, \dots, v_{l-1})$ e $\boldsymbol{v}_{>l} = (v_{l+1}, \dots, v_n)$. Empregando esta notación, a función de densidade $f : \mathbb{R}^n \longrightarrow \mathbb{R}^+$ para unha rede autorregresiva como NADE pódese

escribir da seguinte maneira:

$$f(\boldsymbol{v}) = \prod_{i=1}^{n} f(v_i \mid \boldsymbol{v}_{<i}). \tag{3.1}$$

Estudaremos o modelo orixinal de NADE con variables binarias, quedando a extensión ao caso continuo para a Sección 3.3. Por simplicidade, denotamos á función masa de probabilidade como f igualmente. A continuación, veremos como a parametrización das $f(\cdot \mid \boldsymbol{v}_{<i})$ que propón NADE se orixina de forma natural a partir da aplicación dun procedemento de campo medio (Sección 2.1.6) sobre a RBM.

3.1. Parametrización de NADE

Tomemos unha RBM (Sección 2.1.7) con m variables visibles $\boldsymbol{V} = (V_1, \dots, V_m)$, n variables latentes $\boldsymbol{H} = (H_1, \dots, H_n)$ e poñamos que queremos expresar a súa función de densidade na forma da ecuación (3.1), como se fose unha rede autorregresiva. Seguindo as ecuacións da RBM, obtemos:

$$f(\boldsymbol{v}) = \prod_{i=1}^{m} f\left(v_i \mid \boldsymbol{v}_{<i}\right) \tag{3.2}$$

$$= \prod_{i=1}^{m} \frac{f\left(v_i, \boldsymbol{v}_{<i}\right)}{f\left(\boldsymbol{v}_{<i}\right)} \tag{3.3}$$

$$= \prod_{i=1}^{m} \frac{\sum_{\boldsymbol{v}_{>i}, \boldsymbol{h}} e^{-E(\boldsymbol{v}, \boldsymbol{h})}}{\sum_{v_i, \boldsymbol{v}_{>i}, \boldsymbol{h}} e^{-E(\boldsymbol{v}, \boldsymbol{h})}}. \tag{3.4}$$

Non obstante, os sumatorios da ecuación (3.4) son moi custosos de calcular, como vimos na Sección 2.1.7. Uria et al. propoñen empregar o procedemento de campo medio (Sección 2.1.6) para aproximar a función de densidade $f(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i})$. Para entender o motivo desta elección, lembremos que o procedemento consiste en aproximar unha densidade multidimensional asumindo que se descompón como o produto das densidades unidimensionais das variables que a conforman. É dicir:

$$f(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}) \approx q(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}) = q(v_i \mid \boldsymbol{v}_{<i}) \prod_{j=i+1}^{m} q(v_j \mid \boldsymbol{v}_{<i}) \prod_{j=1}^{n} q(h_j \mid \boldsymbol{v}_{<i}).$$

Ao ser variables binarias, as densidades unidimensionais só poden ser a función masa de probabilidade dunha distribución Bernoulli cun parámetro libre:

$$q(v_j \mid \boldsymbol{v}_{<i}) = \xi_j(i)^{v_j}(1 - \xi_j(i))^{1-v_j},$$
$$q(h_j \mid \boldsymbol{v}_{<i}) = \tau_j(i)^{h_j}(1 - \tau_j(i))^{1-h_j},$$

onde $\xi_j(i) \in [0, 1]$ e $\tau_j(i) \in [0, 1]$ son parámetros descoñecidos das Bernoulli que dependen de i e de j. Nótese que a expresión que nos interesa é a de $q(v_j \mid \boldsymbol{v}_{<i})$, que é función dos $\xi_j(i)$, e

nótese tamén que pese a parecer que estamos estimando termos que non necesitamos (como os $q(h_j \mid \boldsymbol{v}_{<i})$), a elección dunha q con todos os termos permítenos omitir o cálculo de custosas distribucións marxinais.

Neste punto, entra en xogo a chave do procedemento variacional. Denotemos como $\hat{\boldsymbol{X}}_i = (V_i, V_{i+1}, \dots, V_m, H_1, \dots, H_n)$ á variable aleatoria multidimensional definida pola masa de probabilidade $f(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i})$ e, analogamente, como $\hat{\boldsymbol{X}}'_i = (V'_i, V'_{i+1}, \dots, V'_m, H'_1, \dots, H'_n)$ á variable aleatoria correspondente a $q(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i})$. Sen facer ningunha hipótese sobre os ξ e os τ, a súa expresión obtense analiticamente sen máis que minimizar a diverxencia KL entre $\hat{\boldsymbol{X}}'_i$ e $\hat{\boldsymbol{X}}_i$. Comecemos calculando a diverxencia:

$$
\begin{aligned}
D_{\mathrm{KL}}\left(\hat{\boldsymbol{X}}'_i \| \hat{\boldsymbol{X}}_i\right) =& \mathbb{E}_{\hat{\boldsymbol{X}}'_i}\left[\log(q(V'_i, \boldsymbol{V}'_{>i}, \boldsymbol{H}' \mid \boldsymbol{V}'_{<i})) - \log(f(V'_i, \boldsymbol{V}'_{>i}, \boldsymbol{H}' \mid \boldsymbol{V}'_{<i}))\right] \\
=& -\sum_{v_i, \boldsymbol{v}_{>i}, \boldsymbol{h}} q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \log f\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \\
&+ \sum_{v_i, \boldsymbol{v}_{>i}, \boldsymbol{h}} q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \log q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \\
\text{(ver Apéndice A)} =& \log\left(\sum_{v_i, \boldsymbol{v}_{>i}, \boldsymbol{h}} e^{-E(\boldsymbol{v}, \boldsymbol{h})}\right) - \sum_{j=1}^{m}\sum_{k=1}^{n} \tau_k(i) W_{k,j} \xi_j(i) - \sum_{j=1}^{m} b_j \xi_j(i) - \sum_{k=1}^{n} c_k \tau_k(i) \\
&+ \sum_{j=i}^{m} \left(\xi_j(i) \log \xi_j(i) + (1 - \xi_j(i)) \log\left(1 - \xi_j(i)\right)\right) \\
&+ \sum_{k=1}^{n} \left(\tau_k(i) \log \tau_k(i) + (1 - \tau_k(i)) \log\left(1 - \tau_k(i)\right)\right).
\end{aligned}
\tag{3.5}
$$

Na última igualdade, por simplicidade, $\xi_j(i) = v_j$ para $j < i$. Lembremos que os escalares $\tau_k(i) \in [0, 1]$ non son máis que parámetros das distribucións Bernoulli e, polo tanto, podemos fixar k e i e tomar a derivada respecto a $\tau_k(i)$:

$$
\frac{\partial D_{\mathrm{KL}}\left(\hat{\boldsymbol{X}}'_i \| \hat{\boldsymbol{X}}_i\right)}{\partial \tau_k(i)} = -c_k - \sum_{j=1}^{m} W_{k,j} \xi_j(i) + \log\left(\frac{\tau_k(i)}{1 - \tau_k(i)}\right).
$$

Igualando a 0 e despexando obtemos un único punto crítico:

$$
0 = -c_k - \sum_{j=1}^{m} W_{k,j} \xi_j(i) + \log\left(\frac{\tau_k(i)}{1 - \tau_k(i)}\right) \tag{3.6}
$$

$$
\frac{\tau_k(i)}{1 - \tau_k(i)} = \exp\left\{c_k + \sum_{j=1}^{m} W_{k,j} \xi_j(i)\right\} \tag{3.7}
$$

$$
\tau_k(i) = \frac{\exp\left\{c_k + \sum_{j=1}^{m} W_{k,j} \xi_j(i)\right\}}{1 + \exp\left\{c_k + \sum_{j=1}^{m} W_{k,j} \xi_j(i)\right\}} \tag{3.8}
$$

$$
\tau_k(i) = \sigma\left(c_k + \sum_{j=i}^{m} W_{k,j} \xi_j(i) + \sum_{j=1}^{i-1} W_{k,j} v_j\right). \tag{3.9}
$$

Este punto crítico será necesariamente un mínimo, xa que a derivada é monótona crecente respecto a $\tau_k(i)$.

Analogamente á dedución que acabamos de realizar para $\tau_k(i)$, derivemos a diverxencia KL respecto a $\xi_j(i)$ e igualemos a 0:

$$0 = \frac{\partial D_{\mathrm{KL}}\left(\hat{\boldsymbol{X}}'_i \| \hat{\boldsymbol{X}}_i\right)}{\partial \xi_j(i)}, \tag{3.10}$$

$$0 = -b_j - \sum_{k=1}^{n} \tau_k(i) W_{k,j} + \log\left(\frac{\xi_j(i)}{1-\xi_j(i)}\right), \tag{3.11}$$

$$\frac{\xi_j(i)}{1-\xi_j(i)} = \exp\left\{b_j + \sum_{k=1}^{n} \tau_k(i) W_{k,j}\right\}, \tag{3.12}$$

$$\xi_j(i) = \frac{\exp\left\{b_j + \sum_{k=1}^{n} \tau_k(i) W_{k,j}\right\}}{1 + \exp\left\{b_j + \sum_{k=1}^{n} \tau_k(i) W_{k,j}\right\}}, \tag{3.13}$$

$$\xi_j(i) = \sigma\left(b_j + \sum_{k=1}^{n} \tau_k(i) W_{k,j}\right). \tag{3.14}$$

O procedemento variacional completo sobre unha RBM consistiría en resolver o sistema que conforman as ecuacións (3.9) e (3.14) que, ao depender unha da outra, resolveríanse mediante un procedemento de punto fixo, aplicándoas alternativamente. Non obstante, xa adiantamos que NADE non é unha RBM, senón que é unha rede autorregresiva que se inspira na RBM para parametrizar os factores da ecuación (3.2). En concreto, os autores propoñen realizar soamente unha iteración do procedemento de punto fixo, empregando a ecuación (3.9) cos $\xi_j(i)$ inicializados a 0. Pasando xa a notación de NADE as ecuacións son:

$$\boldsymbol{h}_i = \sigma(\boldsymbol{W}_{\cdot,<i}\boldsymbol{v}_{<\boldsymbol{i}} + \boldsymbol{c}), \tag{3.15}$$

$$\mathbb{P}(V_i = 1 \mid \boldsymbol{v}_{<i}) = \sigma(\hat{\boldsymbol{W}}_{i,\cdot}\boldsymbol{h}_i + b_i), \tag{3.16}$$

$$f(\boldsymbol{v}) = \prod_{i=1}^{n} f(v_i \mid \boldsymbol{v}_{<i}) = \prod_{i=1}^{n} \mathbb{P}(V_i = 1 \mid \boldsymbol{v}_{<i})^{v_i}(1 - \mathbb{P}(V_i = 1 \mid \boldsymbol{v}_{<i}))^{1-v_i}. \tag{3.17}$$

Nótese que os vectores $\boldsymbol{h}_i$ fan o papel dos $\tau(i)$ e que $\mathbb{P}(V_i = 1 \mid \boldsymbol{v}_{<i})$ deriva de $\xi(i)$. Ademais, Uria et al. decidiron introducir unha matriz $\hat{\boldsymbol{W}}$ para o cálculo das probabilidades, ao observar empiricamente que desligar estes parámetros melloraba o rendemento (na formulación da RBM, $\boldsymbol{W}$ fai tamén o papel de $\hat{\boldsymbol{W}}$). Deste xeito, as ecuacións (3.15) e (3.16) corresponden á aplicación dunha rede neuronal cunha capa oculta, tal e como vimos na Sección 1.3. Isto permítenos aproveitar todas as ferramentas existentes para redes neuronais, con librerías como *Theano*, presentada en Al-Rfou et al. (2016), que executan a multiplicación de matrices concorrentemente en aceleradores gráficos e proporcionan diferenciación automática para o proceso de adestramento.

A maiores, a formulación é eficiente no número de parámetros, ao compartir a matriz $\boldsymbol{W}$ entre os distintos $\boldsymbol{h_i}$. É dicir, tal e como vemos na ecuación (3.15), segundo vai aumentando

o índice i, vanse empregando máis columnas da matriz $\boldsymbol{W}$. Isto permite reducir o número de parámetros de NADE dende a orde $O(m^2)$ de LARC ata $O(nm)$ (lembremos que m é o número de variables visibles $\boldsymbol{V}$ e n o de variables latentes $\boldsymbol{H}$). Como o índice i é incremental, o cálculo dos $\boldsymbol{h}_i$ pódese implementar moi eficientemente a partir do cálculo de h_{i-1}, sen máis que introducir un vector intermedio que denotaremos como $\boldsymbol{a}_i$:

$$\begin{aligned}
&\boldsymbol{h}_1 = \sigma\left(\boldsymbol{a}_1\right), \quad \text{sendo } \boldsymbol{a}_1 = \boldsymbol{c}, \\
&\boldsymbol{h}_i = \sigma\left(\boldsymbol{a}_i\right), \quad \text{sendo } \boldsymbol{a}_i = \boldsymbol{W}_{\cdot,<i}\boldsymbol{v}_{<i} + \boldsymbol{c} = \boldsymbol{W}_{\cdot,i-1}v_{i-1} + \boldsymbol{a}_{i-1}, \quad \text{para } i = 2, \dots, m.
\end{aligned}$$

Con este cálculo intermedio, a complexidade computacional de NADE para calcular $f(\boldsymbol{v})$ é tamén de $O(mn)$. A Figura 3.1 amosa un esquema dos pasos necesarios para este cómputo:

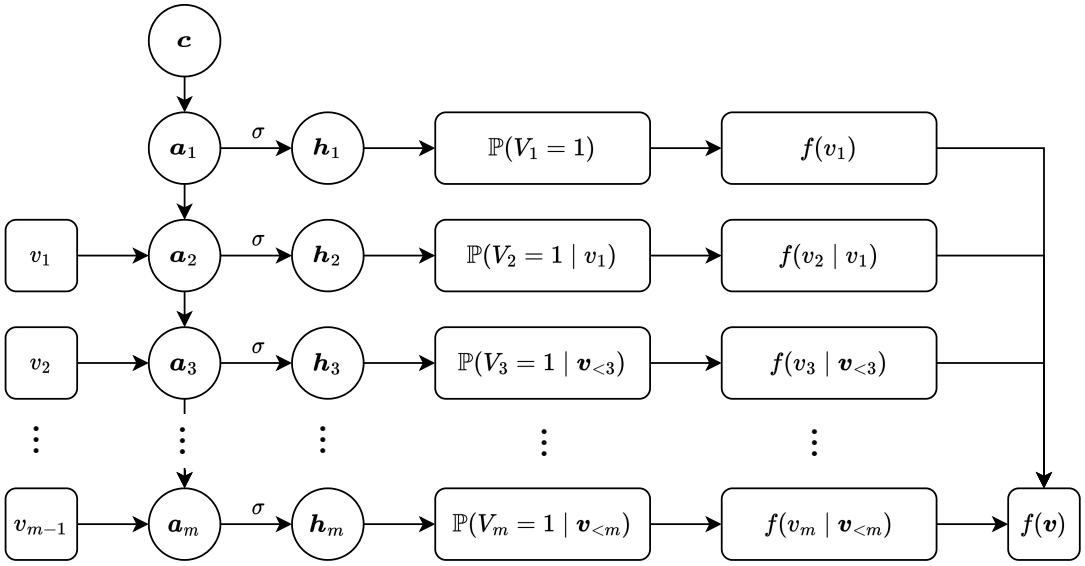


Figura 3.1: Esquema da computación dun vector $\boldsymbol{v}$ no modelo NADE. Os rectángulos indican escalares, e o cómputo sucede de arriba a abaixo e de esquerda a dereita.

3.2. Implementación de NADE

O algoritmo de axuste da RBM necesitaba da simulación de mostras, e esta realizábase mediante o custoso algoritmo de Gibbs. No caso de NADE, o feito de que $f(\boldsymbol{v})$ consista en simples sumas e multiplicacións vectoriais facilita enormemente estas tarefas e, ademais, proporciona de forma directa a verosimilitude dun $\boldsymbol{v}$ dado.

Para axustar os parámetros, aplícase de novo unha optimización numérica da función de verosimilitude, tal e como faciamos na Sección 2.1.5. Neste caso, dada unha mostra $\boldsymbol{v}^{(1)}, \dots, \boldsymbol{v}^{(M)}$,

trátase de maximizar:

$$\sum_{i=1}^{M} \log f_{\boldsymbol{\theta}}\left(\boldsymbol{v}^{(i)}\right) = \sum_{i=1}^{M}\sum_{j=1}^{m} \log f_{\boldsymbol{\theta}}\left(v_j^{(i)} \mid \boldsymbol{v}_{<j}^{(i)}\right). \tag{3.18}$$

A diferenciación $\nabla_{\boldsymbol{\theta}} \log f_{\boldsymbol{\theta}}(\boldsymbol{v})$ é moi sinxela e nas implementacións modernas é realizada automaticamente por ferramentas de redes neuronais como *PyTorch*.

Para xerar mostras con NADE, só hai que modificar lixeiramente o procedemento amosado na Figura 3.1, empregando o método ancestral presentado na Sección 2.2. Primeiro, obtense $\mathbb{P}(V_1 = 1)$ do modelo, que é o parámetro dunha Bernoulli que empregamos para simular v_1. A continuación, calcúlase $\mathbb{P}(V_2 = 1 \mid v_1)$ para simular do mesmo xeito v_2. Tras repetir este procedemento ata obter v_m, NADE é capaz de simular mostras $\boldsymbol{v}$ independentes que seguen a distribución axustada polo modelo.

3.3. Extensión a un dominio real: RNADE

Ata o de agora tratamos o caso de variables binarias. Vexamos un exemplo de extensión a outros dominios, a través do modelo baseado en NADE presentado en Uria et al. (2013).

Hai varias alternativas para estender un modelo como NADE a datos reais. Unha aproximación *naive* (sen garantía teórica) sería empregar directamente o valor de $\mathbb{P}(V_i = 1 \mid \boldsymbol{v}_{<i})$ como saída da rede. Outra posibilidade é partir dunha RBM adaptada a dominios continuos, como por exemplo a RBM baseada en distribucións normais de varianza fixa que propoñen en Welling et al. (2004). A partir desta RBM pódense derivar novas fórmulas de NADE seguindo o procedemento da Sección 3.1. Así, o modelo resultante produciría a media dunha distribución normal con varianza fixa para cada condicional da ecuación 3.17. Estas dúas aproximacións foron probadas experimentalmente en Uria et al. (2013), sen resultados competitivos. Finalmente, esa mesma publicación propón o seguinte enfoque: cada condicional da ecuación (3.17) modelarase mediante unha mestura de K normais unidimensionais (véxase a Sección 1.1.2). Deste xeito, o modelo RNADE deberá producir os pesos da mestura, o vector de medias e o vector de desviacións típicas:

$$\begin{aligned}
&f(v_i \mid \boldsymbol{v}_{<i}) = \sum_{k=1}^{K} \pi_{i,k} N(v_i; \mu_{i,k}, \sigma_{i,k}^2), \text{ con } \sum_{k=1}^{K} \pi_k = 1, \\
&\pi_{i,k} = \frac{e^{z_{i,k}^{(\pi)}}}{\sum_{k=1}^{K} e^{z_{i,k}^{(\pi)}}}, \quad z_{i,k}^{(\pi)} = b_{i,k}^{(\pi)} + \hat{\boldsymbol{W}}_{i,\cdot}^{(\pi,k)} \boldsymbol{h}_i, \\
&\mu_{i,k} = z_{i,k}^{(\mu)}, \qquad\qquad z_{i,k}^{(\mu)} = b_{i,k}^{(\mu)} + \hat{\boldsymbol{W}}_{i,\cdot}^{(\mu,k)} \boldsymbol{h}_i, \\
&\sigma_{i,k} = e^{z_{i,k}^{(\sigma)}}, \qquad\qquad z_{i,k}^{(\sigma)} = b_{i,k}^{(\sigma)} + \hat{\boldsymbol{W}}_{i,\cdot}^{(\sigma,k)} \boldsymbol{h}_i.
\end{aligned}$$

Segundo se desprende da parametrización, o modelo require tres matrices de pesos $\hat{\boldsymbol{W}}$ por cada compoñente da mestura: unha para os pesos $\pi_{i,k}$, outra para as medias $\mu_{i,k}$ e outra máis para as desviacións típicas $\sigma_{i,k}$.

3.4. Outras variantes: DeepNADE, EoNADE, NADE-k

Ademais desta ampliación a un dominio real, propuxéronse outras extensións de NADE. Uria et al. (2014) incrementaron a profundidade das redes neuronais, engadindo máis capas ocultas nun modelo que denominaron DeepNADE. Tamén abordaron posibles estratexias para paliar un dos maiores defectos de NADE: a dependencia do modelo da ordenación escollida para o vector de entrada $\boldsymbol{v}$ (na Sección 3.6 discutiremos a influencia das ordenacións en NADE). Aumentar a profundidade non mellorou o rendemento significativamente, pero combinar varios modelos adestrados con diferentes ordenacións de $\boldsymbol{v}$ sí mellorou os resultados. Esta arquitectura que pondera varios modelos DeepNADE con ordenacións diferentes denomínase Ensembles of NADEs (EoNADE).

Outro estudo baseado en NADE de moito interese foi NADE-k, presentado por Raiko et al. (2014). Este propón continuar o procedemento variacional da Sección 3.1 ata resolver completamente o sistema de ecuacións formulado, realizando todos os pasos necesarios do método de punto fixo.

3.5. Resultados

Volvamos ao conxunto de datos do MNIST e comparemos NADE cos experimentos da RBM que presentamos na Sección 2.1.7. Executamos unha implementación de NADE na librería de aceleración gráfica *Theano*, adestrando o modelo en 75 pasadas sobre o conxunto de datos con $\lambda = 0{,}001$ e 500 variables latentes. NADE consegue igualar as capacidades xerativas da RBM, ao tempo que baixa o custo computacional e obtén de forma exacta a estimación do modelo para a función de densidade. Na práctica, estas melloras significan que NADE é capaz de xerar as 100 mostras en 0,3 segundos, mentres que a RBM necesita un minuto por díxito. Na Figura 3.2 obsérvase como as mostras simuladas con NADE están mellor distribuídas e son máis variadas cás da RBM, ao non sofrer do fenómeno da mestura pobre das cadeas de Markov.

Porén, comparar a calidade visual das mostras xeradas non é unha forma precisa de avaliar os modelos. Para ter unha medida numérica, aproveitaremos que o conxunto de datos MNIST proporciona unha partición de datos de *test* con 10.000 imaxes a maiores das 60.000 que se empregan para axustar os parámetros. Calcularemos a función de verosimilitude destas 10.000

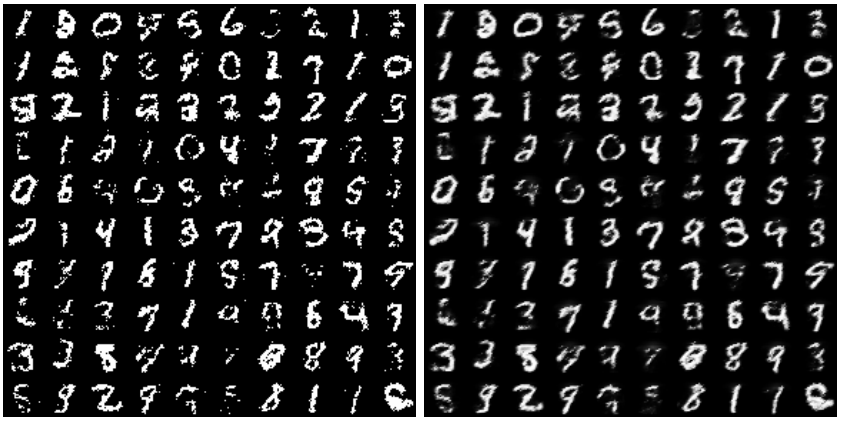


Figura 3.2: Mostras simuladas por NADE. Na esquerda, os píxeles binarios simulados polo modelo. Na dereita, unha representación das probabilidades xeradas para cada píxel. A calidade das mostras na nosa implementación é lixeiramente inferior ás presentadas en Uria et al. (2016), probablemente debido a que os detalles e hiperparámetros da implementación orixinal non son públicos.

imaxes que o modelo nunca viu durante o adestramento. Canto máis alta sexa, máis probabilidade asigna ás imaxes reais do conxunto de datos, e polo tanto mellor axusta o modelo a distribución real dos datos.

No Cadro 3.1 compárase NADE con varios modelos máis simples, como mesturas de Bernoullis ou o exitoso modelo LARC da Sección 1.4. Para a RBM, emprégase unha estimación da verosimilitude baseada en técnicas Monte Carlo. Vemos como a RBM supera amplamente a bondade do axuste destes modelos e observamos como NADE, pese a todas as simplificacións que o fan tratable e sinxelo de operar, obtén uns resultados moi similares. Os datos de verosimilitude foron extraídos de Uria et al. (2016), salvo os da RBM e de NADE, que se obtiveron directamente da nosa simulación e confirman os resultados presentados na publicación.

Nas últimas filas do Cadro 3.1 amósanse os resultados da extensión posterior de NADE presentada en Uria et al. (2014). DeepNADE, un modelo baseado en redes neuronais máis profundas que admite calquera ordenación da secuencia de entrada, obtén peores resultados. Pola contra, a avaliación ponderada de 128 execucións de DeepNADE con diferentes ordenamentos do vector de entrada $\boldsymbol{v}$ si logra superar as aproximacións anteriores.

3.6. Discusión de NADE e conclusións

A modo de conclusión, expoñeremos as principais fortalezas e debilidades de NADE.

Modelo	log f
Mestura Bernoullis K=10	-168.95
Mestura Bernoullis K=500	-137.64
LARC	-97.45
RBM n=500	**-86.54***
NADE n=500	**-88.60**
DeepNADE 4 capas	-89.60
EoNADE 2 capas, 128 ordenacións	**-85.10**

Cadro 3.1: Resultados de verosimilitude de NADE e outros modelos no conxunto de datos MNIST. O valor da RBM(*) non é totalmente comparable, ao ser unha estimación con técnicas Monte Carlo. Datos reproducidos da publicación Uria et al. (2016), salvo os de NADE e da RBM, que proveñen dunha simulación propia.

A principal debilidade, como adiantamos na Sección anterior, é a dependencia que amosa dun ordenamento concreto para o vector de variables visibles $\boldsymbol{v}$. Nalgunhas aplicacións terá sentido fixar unha certa orde das variables pero noutras, como vimos co conxunto MNIST, este "aplanamento" das imaxes é completamente arbitrario. Ademais, tamén determina o subconxunto de densidades condicionais que pode calcular NADE de forma tratable. En Uria et al. (2014) estúdase este problema. Os autores chegan á conclusión de que non hai unha diferencia significativa de rendemento ao seleccionar ordenamentos distintos para $\boldsymbol{v}$. Propoñen un método baseado en DeepNADE (Sección 3.4) que elimina esta restrición, pero perdendo o sustento teórico da RBM. Esta aproximación, como vimos no Cadro 3.1, non alcanza os mellores resultados. Porén, tal como adiantamos na Sección 3.5, DeepNADE obtén un rendemento competitivo en estimación de densidade cando avalía conxuntamente moitos ordenamentos do vector de variables visibles.

Ademais desta primeira debilidade, realizar só unha iteración do algoritmo de punto fixo que resolve as ecuacións do procedemento variacional é unha decisión de carácter práctico moi pouco satisfactoria dende o punto de vista teórico. O mesmo sucede coa separación da matriz de pesos en $\boldsymbol{W}$ e $\hat{\boldsymbol{W}}$. Raiko et al. (2014) abordan esta cuestión, adestrando variantes de NADE que realizan ata 5 iteracións do algoritmo de punto fixo. Con todo, non logra ningunha mellora significativa ao resultado de NADE do Cadro 3.1, a non ser que xunten esta aproximación coas melloras de Uria et al. (2014) presentadas no parágrafo anterior.

Pese a estas dúas cuestións, NADE consegue modelar con éxito datos de moi alta dimensión, posibilitando a estimación de densidade dun xeito computacionalmente tratable e sen perder rendemento nin capacidade de xeneralización respecto á RBM. Os traballos posteriores revisaron as principais vías de melloras de NADE, pero aínda quedan outras liñas de investigación abertas. Por exemplo, LARC e NADE parten dunha rede autorregresiva para garantir a tratabilidade

computacional. Outros modelos gráficos poden proporcionar as mesmas vantaxes computacionais sen impoñer unha orde tan estrita sobre o vector de variables. Outra opción de investigación é a busca doutros mecanismos prácticos que permitan aplicar modelos gráficos a datos nun soporte continuo de xeito eficiente. As aproximacións habituais na literatura, baseadas en mesturas de distribucións normais, non son completamente satisfactorias.

En resumo, neste traballo analizáronse as bases dos modelos gráficos, unha ferramenta estatística para datos estruturados e de alta dimensión. NADE serviu como caso de estudo e nexo de unión entre os principais conceptos relacionados co deseño, axuste e operación destes modelos. Realizouse unha actualización das implementacións destes modelos a código moderno e comprobouse experimentalmente mediante simulacións o seu potencial e os desafíos que presentan, empregando o conxunto de datos MNIST como marco de referencia. Finalmente, observamos como as redes neuronais, un paradigma de plena actualidade no eido da computación, emerxen de forma natural da formulación teórica de varios destes modelos estatísticos.

Apéndice A

Derivación das ecuacións de NADE

Dado que nos cálculos da dedución das fórmulas de NADE a partir da aproximación de campo medio aparecía unha igualdade que non está probada en ningunha das publicacións, neste Apéndice realizarase a devandita demostración.

Situémonos na ecuación (3.5) da Sección 3.1, durante o cálculo de diverxencia de Kullback-Leibler entre a distribución empregada para a aproximación e a distribución da RBM:

$$
\begin{aligned}
D_{\mathrm{KL}}\left(\hat{\boldsymbol{X}}'_i \| \hat{\boldsymbol{X}}_i\right) =& \mathbb{E}_{\hat{\boldsymbol{X}}'_i}\left[\log(q(V'_i, \boldsymbol{V}'_{>i}, \boldsymbol{H}' \mid \boldsymbol{V}'_{<i})) - \log(f(V'_i, \boldsymbol{V}'_{>i}, \boldsymbol{H}' \mid \boldsymbol{V}'_{<i}))\right] \\
=& -\sum_{v_i, \boldsymbol{v}_{>i}, \boldsymbol{h}} q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \log f\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \\
&+ \sum_{v_i, \boldsymbol{v}_{>i}, \boldsymbol{h}} q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \log q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \\
(*) =& \log\left(\sum_{v_i, \boldsymbol{v}_{>i}, \boldsymbol{h}} e^{-E(\boldsymbol{v}, \boldsymbol{h})}\right) - \sum_{j=1}^{m}\sum_{k=1}^{n} \tau_k(i) W_{k,j} \xi_j(i) - \sum_{j=1}^{m} b_j \xi_j(i) - \sum_{k=1}^{n} c_k \tau_k(i) \\
&+ \sum_{j=i}^{m} \left[\xi_j(i) \log \xi_j(i) + (1 - \xi_j(i)) \log\left(1 - \xi_j(i)\right)\right] \\
&+ \sum_{k=1}^{n} \left[\tau_k(i) \log \tau_k(i) + (1 - \tau_k(i)) \log\left(1 - \tau_k(i)\right)\right].
\end{aligned}
$$

Recordemos que $\hat{\boldsymbol{X}}_i = (V_i, V_{i+1}, \ldots, V_m, H_1, \ldots, H_n)$ e $\hat{\boldsymbol{X}}'_i = (V'_i, V'_{i+1}, \ldots, V'_m, H'_1, \ldots, H'_n)$ son vectores aleatorios con densidades asociadas $f(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i})$ e $q(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i})$, co índice i fixado. Probaremos a igualdade $(*)$, dividindo a proba en dúas partes. Por unha banda,

escribimos a primeira igualdade:

$$\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}} q\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)\log f\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)$$
$$=-\log\left(\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}} e^{-E(\boldsymbol{v},\boldsymbol{h})}\right)+\sum_{j=1}^{m}\sum_{k=1}^{n}\tau_k(i)W_{k,j}\xi_j(i)+\sum_{j=1}^{m}b_j\xi_j(i)+\sum_{k=1}^{n}c_k\tau_k(i). \quad \text{(A.1)}$$

Por outra banda, a segunda igualdade sería:

$$\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}} q\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)\log q\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)=\sum_{j=i}^{m}\left[\xi_j(i)\log\xi_j(i)+(1-\xi_j(i))\log\left(1-\xi_j(i)\right)\right]$$
$$+\sum_{k=1}^{n}\left[\tau_k(i)\log\tau_k(i)+(1-\tau_k(i))\log\left(1-\tau_k(i)\right)\right]. \quad \text{(A.2)}$$

Primeira igualdade

Comecemos observando que $f\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)$ é unha función masa de probabilidade condicionada nalgunhas das variables do vector aleatorio $(\boldsymbol{V},\boldsymbol{H})=(V_1,\ldots,V_m,H_1,\ldots,H_n)$ sobre o que está definida f. Lembremos que a masa de probabilidade da máquina de Boltzmann restrinxida defínese en base a unha función de enerxía e á constante de normalización Z (Sección 2.1.1). Polo tanto:

$$\log f\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)=\log\frac{f(\boldsymbol{v},\boldsymbol{h})}{f(\boldsymbol{v}_{<i})}=\log\left(\frac{\frac{e^{-E(\boldsymbol{v},\boldsymbol{h})}}{Z}}{\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}}\frac{e^{-E(\boldsymbol{v},\boldsymbol{h})}}{Z}}\right)$$
$$=-E(\boldsymbol{v},\boldsymbol{h})-\log\left(\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}} e^{-E(\boldsymbol{v},\boldsymbol{h})}\right).$$

A partir deste punto, por simplicidade e pola semellanza coa constante de normalización Z, denotaremos $Z'(\boldsymbol{v}_{<i}):=\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}} e^{-E(\boldsymbol{v},\boldsymbol{h})}$. Aplicamos a función de enerxía da máquina de Boltzmann restrinxida (Sección 2.1.7):

$$\log f\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)=-\boldsymbol{b}^\top\boldsymbol{v}-\boldsymbol{c}^\top\boldsymbol{h}-\boldsymbol{h}^\top\boldsymbol{W}\boldsymbol{v}-\log Z'(\boldsymbol{v}_{<i})$$
$$=\sum_{j=1}^{m}b_jv_j+\sum_{j=1}^{n}c_jh_j+\sum_{j=1}^{m}\sum_{k=1}^{n}h_kW_{k,j}v_j-\log Z'(\boldsymbol{v}_{<i}). \quad \text{(A.3)}$$

Fixémonos agora en q. Por definición, temos:

$$q\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)=\prod_{j=i}^{m}\left[\xi_j(i)^{v_j}(1-\xi_j(i))^{1-v_j}\right]\prod_{j=1}^{n}\left[\tau_j(i)^{h_j}(1-\tau_j(i))^{1-h_j}\right]. \quad \text{(A.4)}$$

Xa estamos en condicións de abordar a primeira igualdade. Substituamos en (A.1) o obtido en (A.4) e separemos os sumatorios, tendo en conta que son variables binarias (os posibles valores son 0 e 1):

$$\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}} q\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)\log f\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)=\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}}\left\{\prod_{j=i}^{m}\left[\xi_j(i)^{v_j}(1-\xi_j(i))^{1-v_j}\right]\right.$$
$$\left.\prod_{j=1}^{n}\left[\tau_j(i)^{h_j}(1-\tau_j(i))^{1-h_j}\right]\log f\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)\right\}$$

$$=\sum_{v_i=0}^{1}\xi_i(i)^{v_i}(1-\xi_i(i))^{1-v_i}\sum_{v_{i+1}=0}^{1}\xi_{i+1}(i)^{v_{i+1}}(1-\xi_{i+1}(i))^{1-v_{i+1}}\sum_{v_{i+2}=0}^{1}\cdots\sum_{v_m=0}^{1}\xi_m(i)^{v_m}(1-\xi_m(i))^{1-v_m}$$
$$\sum_{h_1=0}^{1}\tau_1(i)^{h_1}(1-\tau_1(i))^{1-h_1}\cdots\underbrace{\sum_{h_n=0}^{1}\tau_n(i)^{h_n}(1-\tau_n(i))^{1-h_n}\log f\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)}_{(**)}. \quad \text{(A.5)}$$

Desenvolvamos o fragmento de (A.5) marcado como $(**)$ engadindo o obtido na ecuación (A.3):

$$\begin{aligned}(**)=&\sum_{h_n=0}^{1}\tau_n(i)^{h_n}(1-\tau_n(i))^{1-h_n}\left[\sum_{j=1}^{m}b_jv_j+\sum_{j=1}^{n}c_jh_j+\sum_{j=1}^{m}\sum_{k=1}^{n}h_kW_{k,j}v_j-\log Z'(\boldsymbol{v}_{<i})\right]\\
=&\tau_n(i)\left[c_n+\sum_{j=1}^{m}W_{n,j}v_j+\sum_{j=1}^{m}b_jv_j+\sum_{j=1}^{n-1}c_jh_j+\sum_{j=1}^{m}\sum_{k=1}^{n-1}h_kW_{k,j}v_j-\log Z'(\boldsymbol{v}_{<i})\right]\\
&+(1-\tau_n(i))\left[\sum_{j=1}^{m}b_jv_j+\sum_{j=1}^{n-1}c_jh_j+\sum_{j=1}^{m}\sum_{k=1}^{n-1}h_kW_{k,j}v_j-\log Z'(\boldsymbol{v}_{<i})\right]\\
=&\tau_n(i)\left(c_n+\sum_{j=1}^{m}W_{n,j}v_j\right)+\sum_{j=1}^{m}b_jv_j+\sum_{j=1}^{n-1}c_jh_j+\sum_{j=1}^{m}\sum_{k=1}^{n-1}h_kW_{k,j}v_j-\log Z'(\boldsymbol{v}_{<i}). \quad \text{(A.6)}\end{aligned}$$

Substituimos de volta $(**)$ na ecuación (A.5):

$$\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}} q\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)\log f\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)=\sum_{v_i=0}^{1}\xi_i(i)^{v_i}(1-\xi_i(i))^{1-v_i}\cdots$$
$$\cdots\sum_{v_m=0}^{1}\xi_m(i)^{v_m}(1-\xi_m(i))^{1-v_m}\sum_{h_1=0}^{1}\tau_1(i)^{h_1}(1-\tau_1(i))^{1-h_1}\cdots\sum_{h_{n-1}=0}^{1}\tau_{n-1}(i)^{h_{n-1}}(1-\tau_{n-1}(i))^{1-h_{n-1}}$$
$$\left[\tau_n(i)\left(c_n+\sum_{j=1}^{m}W_{n,j}v_j\right)+\sum_{j=1}^{m}b_jv_j+\sum_{j=1}^{n-1}c_jh_j+\sum_{j=1}^{m}\sum_{k=1}^{n-1}h_kW_{k,j}v_j-\log Z'(\boldsymbol{v}_{<i})\right]. \quad \text{(A.7)}$$

O proceso realizado con (∗∗) na ecuación (A.6) permitiunos eliminar o sumatorio de h_n, como se observa na ecuación (A.7). Debemos repetir este proceso recursivamente sobre (A.7), ata desfacernos de todos os sumatorios das variables $h_1, \dots, h_{n-1}$. Deste xeito, queda:

$$\sum_{v_i, \boldsymbol{v}_{>i}, \boldsymbol{h}} q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \log f\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) = \sum_{v_i=0}^{1} \xi_i(i)^{v_i}(1-\xi_i(i))^{1-v_i} \cdots$$
$$\cdots \sum_{v_m=0}^{1} \xi_m(i)^{v_m}(1-\xi_m(i))^{1-v_m} \left[\sum_{k=1}^{n} \tau_k(i) \left(c_k + \sum_{j=1}^{m} W_{k,j} v_j\right) + \sum_{j=1}^{m} b_j v_j + - \log Z'(\boldsymbol{v}_{<i})\right].$$

A expresión é similar, pero agora cos sumatorios das variables $v_i, v_{i+1}, \dots, v_m$. Repetimos o proceso para eliminar o sumatorio de v_m:

$$\sum_{v_i, \boldsymbol{v}_{>i}, \boldsymbol{h}} q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \log f\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) = \sum_{v_i=0}^{1} \xi_i(i)^{v_i}(1-\xi_i(i))^{1-v_i} \cdots$$
$$\cdots \sum_{v_{m-1}=0}^{1} \xi_{m-1}(i)^{v_{m-1}}(1-\xi_{m-1}(i))^{1-v_{m-1}} \left[\xi_m(i) \left(b_m + \sum_{k=1}^{n} \tau_k(i) W_{k,m}\right)\right.$$
$$\left. + \sum_{k=1}^{n} \tau_k(i) \left(c_k + \sum_{j=1}^{m-1} W_{k,j} v_j\right) + \sum_{j=1}^{m-1} b_j v_j - \log Z'(\boldsymbol{v}_{<i})\right].$$

De novo, repítese recursivamente o proceso ata eliminar todos os sumatorios das variables $v_i, v_{i+1}, \dots, v_{m-1}$. Así, chegamos finalmente á proba da primeira igualdade:

$$\sum_{v_i, \boldsymbol{v}_{>i}, \boldsymbol{h}} q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \log f\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right)$$
$$= \sum_{j=i}^{m} \xi_j(i) \left(b_j + \sum_{k=1}^{n} \tau_k(i) W_{k,j}\right) + \sum_{k=1}^{n} \tau_k(i) \left(c_k + \sum_{j=1}^{i-1} W_{k,j} v_j\right) + \sum_{j=1}^{i-1} b_j v_j - \log Z'(\boldsymbol{v}_{<i})$$
$$= - \log Z'(\boldsymbol{v}_{<i}) + \sum_{j=1}^{m} \sum_{k=1}^{n} \tau_k(i) W_{k,j} \xi_j(i) + \sum_{j=1}^{m} \xi_j(i) b_j + \sum_{k=1}^{n} \tau_k(i) c_k.$$

No último paso empregouse a notación compacta $\xi_j(i) = v_j$ para $j < i$. Isto permite condensar os dous sumatorios que involucran a $\boldsymbol{b}$ e as dúas multiplicacións da matriz $\boldsymbol{W}$.

Segunda igualdade

Comezamos substituíndo o obtido na ecuación (A.4) e separando os sumatorios, igual que fixemos na outra igualdade:

$$\begin{aligned}
&\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}} q\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)\log q\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)\\
&=\sum_{v_i=0}^{1}\xi_i(i)^{v_i}(1-\xi_i(i))^{1-v_i}\sum_{v_{i+1}=0}^{1}\xi_{i+1}(i)^{v_{i+1}}(1-\xi_{i+1}(i))^{1-v_{i+1}}\sum_{v_{i+2}=0}^{1}\cdots\sum_{v_m=0}^{1}\xi_m(i)^{v_m}(1-\xi_m(i))^{1-v_m}\\
&\qquad\sum_{h_1=0}^{1}\tau_1(i)^{h_1}(1-\tau_1(i))^{1-h_1}\cdots\underbrace{\sum_{h_n=0}^{1}\tau_n(i)^{h_n}(1-\tau_n(i))^{1-h_n}\log q\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)}_{(**)}. \qquad \text{(A.8)}
\end{aligned}$$

De novo, operamos co fragmento marcado con $(**)$ introducindo de novo a ecuación (A.4) para eliminar o sumatorio da variable h_n:

$$\begin{aligned}
&\sum_{h_n=0}^{1}\tau_n(i)^{h_n}(1-\tau_n(i))^{1-h_n}\log q\left(v_i,\boldsymbol{v}_{>i},\boldsymbol{h}\mid\boldsymbol{v}_{<i}\right)\\
&=\sum_{h_n=0}^{1}\tau_n(i)^{h_n}(1-\tau_n(i))^{1-h_n}\log\left[\prod_{j=i}^{m}\left(\xi_j(i)^{v_j}(1-\xi_j(i))^{1-v_j}\right)\prod_{j=1}^{n}\left(\tau_j(i)^{h_j}(1-\tau_j(i))^{1-h_j}\right)\right]\\
&=\sum_{h_n=0}^{1}\tau_n(i)^{h_n}(1-\tau_n(i))^{1-h_n}\left[\sum_{j=i}^{m}\log\left(\xi_j(i)^{v_j}(1-\xi_j(i))^{1-v_j}\right)+\sum_{j=1}^{n}\log\left(\tau_j(i)^{h_j}(1-\tau_j(i))^{1-h_j}\right)\right]\\
&=\tau_n(i)\left[\log\tau_n(i)+\sum_{j=i}^{m}\log\left(\xi_j(i)^{v_j}(1-\xi_j(i))^{1-v_j}\right)+\sum_{j=1}^{n-1}\log\left(\tau_j(i)^{h_j}(1-\tau_j(i))^{1-h_j}\right)\right]\\
&\quad+(1-\tau_n(i))\left[(1-\tau_n(i))+\sum_{j=i}^{m}\log\left(\xi_j(i)^{v_j}(1-\xi_j(i))^{1-v_j}\right)+\sum_{j=1}^{n-1}\log\left(\tau_j(i)^{h_j}(1-\tau_j(i))^{1-h_j}\right)\right]\\
&=\tau_n(i)\log\tau_n(i)+(1-\tau_n(i))\log(1-\tau_n(i))+\sum_{j=i}^{m}\log\left[\xi_j(i)^{v_j}(1-\xi_j(i))^{1-v_j}\right]\\
&\qquad\qquad+\sum_{j=1}^{n-1}\log\left(\tau_j(i)^{h_j}(1-\tau_j(i))^{1-h_j}\right).
\end{aligned}$$

Deste xeito elimínase o sumatorio da variable h_n. Repetindo o proceso, na ecuación (A.8) quedan só os sumatorios das variables v_i:

$$\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}} q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \log q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right)$$
$$= \sum_{v_i=0}^{1} \xi_i(i)^{v_i}(1-\xi_i(i))^{1-v_i} \cdots \sum_{v_m=0}^{1} \xi_m(i)^{v_m}(1-\xi_m(i))^{1-v_m}$$
$$\left[\sum_{j=1}^{n} \tau_j(i)\log\tau_j(i) + (1-\tau_j(i))\log(1-\tau_j(i)) + \sum_{j=i}^{m} \log\left(\xi_j(i)^{v_j}(1-\xi_j(i))^{1-v_j}\right)\right].$$

Analogamente, elimínanse os sumatorios das variables v_i, chegando finalmente á segunda igualdade:

$$\sum_{v_i,\boldsymbol{v}_{>i},\boldsymbol{h}} q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right) \log q\left(v_i, \boldsymbol{v}_{>i}, \boldsymbol{h} \mid \boldsymbol{v}_{<i}\right)$$
$$= \sum_{j=i}^{m} \left(\xi_j(i)\log\xi_j(i) + (1-\xi_j(i))\log\left(1-\xi_j(i)\right)\right)$$
$$+ \sum_{k=1}^{n} \left(\tau_k(i)\log\tau_k(i) + (1-\tau_k(i))\log\left(1-\tau_k(i)\right)\right).$$

Así, queda probada a segunda igualdade (A.2) e rematamos a proba da ecuación (3.5).

Bibliografía

Cormen, T., Leiserson, C., Rivest, R. & Stein, C. (2022). *Introduction to Algorithms, 4th Edition*. MIT Press.

Cover, T. M. & Thomas, J. A. (2006). *Elements of Information Theory, 2nd Edition*. Wiley-Interscience.

Fischer, A. & Igel, C. (2012). An Introduction to Restricted Boltzmann Machines. En *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications* (pp. 14–36). Springer.

Frey, B. (1998). *Graphical Models for Machine Learning and Digital Communication*. MIT Press.

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning*. MIT Press.

Jordan, M. I. (2004). Graphical models. *Statistical Science*, *19*(1), 140–155.

Koller, D. & Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

Kulkarni, V. G. (2016). *Modeling and Analysis of Stochastic Systems*. Chapman and Hall/CRC.

LeCun, Y., Cortes, C. & Burges, C. (2010). MNIST handwritten digit database. *ATT Labs*. Dispoñible en http://yann.lecun.com/exdb/mnist. Consultado por última vez o 20/4/2022.

Pinkus, A. (1999). Approximation theory of the MLP model in neural networks. *Acta Numerica*, *8*, 143–195.

Raiko, T., Li, Y., Cho, K. & Bengio, Y. (2014). Iterative Neural Autoregressive Distribution Estimator NADE-k. *Advances in Neural Information Processing Systems*, *27*.

Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., ... Zhang, Y. (2016). Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, *abs/1605.02688*.

Ross, S. (1996). *Stochastic Processes*. Wiley.

Saul, L. & Jordan, M. (1995). Exploiting Tractable Substructures in Intractable Networks. En D. Touretzky, M. Mozer & M. Hasselmo (Eds.), *Advances in Neural Information Processing Systems* (Vol. 8), MIT Press.

Scott, D. (2015). *Multivariate Density Estimation: Theory, Practice, and Visualization*. Wiley.

Smolensky, P. (1986). Information processing in dynamical systems: Foundations of harmony theory. En *Parallel distributed processing: Explorations in the microstructure of cognition* (pp. 194–281). MIT Press.

Uria, B., Côté, M.-A., Gregor, K., Murray, I. & Larochelle, H. (2016). Neural autoregressive distribution estimation. *The Journal of Machine Learning Research*, *17*(1), 7184–7220.

Uria, B., Murray, I. & Larochelle, H. (2013). RNADE: The real-valued neural autoregressive density-estimator. *Advances in Neural Information Processing Systems*, *26*, 2175–2183.

Uria, B., Murray, I. & Larochelle, H. (2014). A deep and tractable density estimator. En *International Conference on Machine Learning* (pp. 467–475). PMLR.

Wand, M. & Jones, M. (1994). *Kernel Smoothing*. Taylor & Francis.

Welling, M., Rosen-Zvi, M. & Hinton, G. (2004). Exponential Family Harmoniums with an Application to Information Retrieval. En *Proceedings of the 17th International Conference on Neural Information Processing Systems* (pp. 1481–1488). MIT Press.