MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

INPE-5570-PRE/1795

CLASSIFICAÇÃO POR MÁXIMA VEROSSIMILHANÇA E POR REDES NEURAIS: UM ESTUDO MONTE CARLO

LÚBIA VINHAS
ALEJANDRO C. FRERY

CONGRESSO NACIONAL DE MATEMÁTICA APLICADA E COMPUTACIONAL, 6., Uberlândia, MG, 6-9 set. 1993.

Resumo das Comunicações, p. 131

INPE
São José dos Campos
1993

Classificação por Máxima Verossimilhança e por Redes Neurais: um Estudo Monte Carlo

Lúbia Vinhas
INPE–UPG/CAP
lubia@pgrad.inpe.br

Alejandro C. Frery
INPE–DPI
frery@dpi.inpe.br

Avenida dos Astronautas, 1758
12227-010 São José dos Campos, SP

Resumo

Neste trabalho compara-se o desempenho de dois métodos para classificação de amostras: o de máxima verossimilhança e o que envolve o uso de uma rede neural.

O problema de classificação de uma observação (em $\mathbb{R}^2$) em uma entre duas classes com treinamento é formulado matematicamente. Dado que, em geral, existe um volume grande de amostras é possível treinar uma rede neural do tipo *backpropagation* para resolver este problema.

Derivam-se algumas propriedades teóricas relevantes ao problema (estimadores e probabilidades de erro) para serem comparadas com os resultados de uma experiência Monte Carlo.

Essa experiência de simulação estocástica consta de três fases:

1. Fase de treinamento: gerar as amostras de treinamento; usá-las como *input* para o classificador por máxima verossimilhança e para o treinamento da rede.
2. Fase de classificação: gerar as amostras a serem classificadas usando novamente os dois métodos.
3. Comparação dos resultados obtidos pelos dois métodos.

Estes passos são repetidos para diferentes tamanhos de amostras e diferentes valores dos parâmetros. Somente foi considerada a distribuição Normal bivariada.

1 Introdução, Notação e o Método de Máxima Verossimilhança

Seja $(\Omega, \mathcal{A}, \mathbb{P}r)$ um espaço de probabilidade. Considera-se $\mathcal{A} = \mathcal{B}(\mathbb{R}^2)$, os Borelianos de $\mathbb{R}^2$. O vetor aleatório $\mathbf{X} = (X_1, X_2): \Omega \longrightarrow \mathbb{R}^2$ tem distribuição normal bivariada com média $\boldsymbol{\mu} = (\mu_1, \mu_2)^{\mathrm{T}} \in \mathbb{R}^2$ e matriz de covariância $\Sigma_{2\times 2}$ se a sua densidade é:

$$f_{\mathbf{X}}(\mathbf{x}) = \frac{1}{2\pi\sqrt{|\Sigma|}} \exp\left((\boldsymbol{\mu} - \mathbf{x})^{\mathrm{T}}\Sigma^{-1}(\boldsymbol{\mu} - \mathbf{x})\right), \tag{1}$$

para todo $\mathbf{x} \in \mathbb{R}^2$. Na equação (1), $|\Sigma|$ denota o determinante da matriz Σ. Denota-se $\mathbf{X} \sim \mathcal{N}^2(\boldsymbol{\mu}, \Sigma)$. Em [12], pode-se ver que para que $\mathbf{X} \sim \mathcal{N}2(\boldsymbol{\mu}, \Sigma)$, Σ tem que ser definida positiva. Reciprocamente, toda matriz definida positiva é matriz de covariância de um vetor $\mathbf{X}$ com distribuição normal bivariada.

O problema aqui abordado consiste em se supor que $\mathbf{x} = (x_1, x_2)^{\mathrm{T}}$ é a ocorrência ou da variável aleatória $\mathbf{Y} \sim \mathcal{N}2(\boldsymbol{\mu}^{(1)}, \Sigma^{(1)})$ ou da variável aleatória $\mathbf{Z} \sim \mathcal{N}2(\boldsymbol{\mu}^{(2)}, \Sigma^{(2)})$. Por simplicidade notacional, escrever-se-á $\mathcal{N}2(\boldsymbol{\mu}^{(\ell)}, \Sigma^{(\ell)})$ como $\mathcal{N}2^{(\ell)}$ para $\ell = 1, 2$. Supõe-se que acontece uma das seguintes situações:

1. $\boldsymbol{\mu}^{(1)} \neq \boldsymbol{\mu}^{(2)}$ e $\Sigma^{(1)} = \Sigma^{(2)}$, ou
2. $\boldsymbol{\mu}^{(1)} = \boldsymbol{\mu}^{(2)}$ e $\Sigma^{(1)} \neq \Sigma^{(2)}$, ou
3. $\boldsymbol{\mu}^{(1)} \neq \boldsymbol{\mu}^{(2)}$ e $\Sigma^{(1)} \neq \Sigma^{(2)}$.

Assim, dado $\mathbf{X}(\omega) = \mathbf{x}$, deseja-se atribuí-lo à "classe" $\ell = 1$ ou à classe $\ell = 2$, tal que seja satisfeito algum critério baseado no "custo" que o experimentador esteja disposto a "pagar" [10].

Um critério muito empregado, consiste em se achar a classe ℓ que maximize a probabilidade *a posteriori*, dada pela equação:

$$\mathbb{P}r(L = \ell \mid \mathbf{X} = \mathbf{x}) = \mathbb{P}r(\mathbf{X} = \mathbf{x} \mid L = \ell)\frac{\mathbb{P}r(L = \ell)}{\mathbb{P}r(\mathbf{X} = \mathbf{x})} \tag{2}$$

como na equação (2) o termo $\mathbb{P}r(\mathbf{X} = \mathbf{x})$ independe de ℓ, na prática procura-se a solução de:

$$\text{máx}^{-1}_{\ell\in\{1,2\}}(\mathbb{P}r(\mathbf{X} = \mathbf{x} \mid L = \ell)\mathbb{P}r(L = \ell)). \tag{3}$$

Se $\mathbb{P}r(L = 1) = \mathbb{P}r(L = 2) = 1/2$, a equação (3) (conhecida com critério MAP: máximo *a posteriori*) simplifica-se ficando:

$$C_{\text{MV}} = \text{máx}^{-1}_{\{\ell\in 1,2\}}\mathbb{P}r(\mathbf{X} = \mathbf{x} \mid L = \ell). \tag{4}$$

O critério dado pela solução da equação (4) é conhecido como "máxima verossimilhança".

Assim sendo, conhecidos os valores de $\boldsymbol{\mu}^{(\ell)}$ e $\Sigma^{(\ell)}$, $\ell = 1, 2$ e dado $\mathbf{x} = (x_1, x_2)$ a classe ℓ que satisfaz (4) é dada por:

$$\begin{cases} \ell = 1 & \text{se } g^{(1)}(\mathbf{x}) \geq g^{(2)}(\mathbf{x}) \\ \ell = 2 & \text{se } g^{(2)}(\mathbf{x}) \geq g^{(1)}(\mathbf{x}), \end{cases} \tag{5}$$

onde:

$$g^{(\ell)}(\mathbf{x}) = |\Sigma^{(\ell)}|^{-1/2} \exp\Big((\boldsymbol{\mu}^{(\ell)} - \mathbf{x})^{\mathrm{T}} \left(\Sigma^{(\ell)}\right)^{-1} (\boldsymbol{\mu}^{(\ell)} - \mathbf{x})\Big).$$

Tal como acima descrito, este critério não produz necessariamente soluções disjuntas, pois se $g^{(1)}(\mathbf{x}) = g^{(1)}(\mathbf{x})$, então $\ell = 1$ e $\ell = 2$, simultaneamente, para esse $\mathbf{x}$. Porém, como a variável aleatória considerada possui densidade, a probabilidade de se observar um evento $\mathbf{x}$ tal que isto aconteça tem probabilidade zero.

Sejam agora C_ℓ, $\ell = 1, 2$, os subconjuntos de $\mathbb{R}^2$ tais que $\mathbf{x} \in C_\ell \Rightarrow C_{\mathrm{MV}} = \ell$. As equações de $C_1 \cap C_2$ para os casos (1), (2) e (3) podem ver-se em [10]. A partir do conhecimento de C_ℓ, $\ell = 1, 2$, podem calcular-se (exatamente) as probabilidades de erro do método de classificação por máxima verossimilhança: sejam $f^{(1)}$ e $f^{(2)}$ as densidades das variáveis aleatórias com distribuição $\mathcal{N}2(\boldsymbol{\mu}^{(1)}, \Sigma^{(1)})$ e $\mathcal{N}2(\boldsymbol{\mu}^{(2)}, \Sigma^{(2)})$, respectivamente. A probabilidade de classificar o padrão $\mathbf{x}$ na classe i, sendo que ele provém da classe j é dada por $\mathbb{P}\mathrm{r}(L = j) \int_{C_i} f^{(j)}$. Estas integrais podem ser obtidas pelo uso de tabelas [4].

As aplicações dessa técnica, e das suas extensões para outras distribuições e/ou para mais de duas classes, são inúmeras. Dentro das mais importantes pode-se citar a taxonomia estatística, o controle de qualidade e o sensoriamento remoto [9].

No que diz respeito às últimas aplicações mencionadas, é importante salientar que muitos sistemas de processamento de imagens (SITIM/SGI [15], SPRING [8], entre outros) provêm algoritmos para tal fim.

As redes neurais também têm sido usadas em problemas de classificação. Ou seja, dada uma entrada, deseja-se que a rede produza uma saída que represente a classe à qual essa entrada pertence. As redes neurais têm sido empregadas com sucesso no reconhecimento de caracteres manuscritos, reconhecimento de voz, reconhecimento de alvos em aplicações militares [13, 14] e classificação em sensoriamento remoto [3] entre outros.

Para se fazer um estudo comparativo teórico sobre estes dois métodos de classificação seria necessário se possuir conhecimento sobre os seguintes assuntos:

- validade do modelo proposto, e
- critério empregado pela rede neural.

Podem ser feitas várias ressalvas a respeito do primeiro ponto, por exemplo:

- ⋆ as distribuições $\mathcal{N}k$ (para imagens multiespectrais de k bandas) empregam-se pela relativa simplicidade de manusseio e porque as regiões homogêneas de imagens ópticas apresentam, muitas vezes, histogramas unimodais e aproximadamente simétricos (por banda). Já ao considerar-se as imagens obtidas com iluminação coerente (não ópticas, por exemplo imagens laser ou de radar de abertura sintética), a hipótese de normalidade fracassa por completo [11];
- ⋆ as imagens de sensoriamento remoto são discretizadas e assumem valores, tipicamente, no conjunto $\{0, .., 256\}^k$ para cada pixel; desta maneira, não é mais válida a hipótese de serem ocorrências de variáveis aleatórias com distribuição $\mathcal{N}k$ (pois, se assim fossem, teriam por contradomínio $\mathbb{R}^k$).

A respeito do critério empregado pelas redes neurais devem ser considerados, entre outros, os seguintes pontos:

- ⋆ sabe-se que são adaptativas: elas podem tomar dados e aprender a partir deles, inferindo soluções. Essa habilidade difere das técnicas padrão de software, porque não dependem do conhecimento *a priori*, por parte de programadores, isto é, não necessitam regras explícitas que descrevam numericamente como obter essas soluções;
- ⋆ podem generalizar: elas podem processar corretamente dados *apenas similares* àqueles usados inicialmente no seu treinamento (o treinamento das redes será discutido mais adiante). Com isso, podem tratar dados imperfeitos ou incompletos, o que é de grande utilidade em aplicações práticas visto que os dados do mundo real são, por natureza, ruidosos;
- ⋆ são não-lineares: podem, assim, capturar relações complexas entre as entradas que lhes são apresentadas.

Mas não se pode descrever facilmente *como* as redes trabalham, ou seja, quais os passos lógicos que levam à solução do problema.

Tal como foi exposto, não é viável uma comparação teórica do desempenho dos métodos aqui considerados. Uma forma alternativa de compará-los é pelo uso de técnicas Monte Carlo. Para detalhes a respeito destas técnicas, uma referências completa é [18].

2 Treinamento dos Algoritmos

Em geral, os valores $\boldsymbol{\mu}^{(\ell)}$ e $\Sigma^{(\ell)}$ não estão disponíveis *a priori*. Procede-se, então, a estimá-los a partir de amostras de treinamento tipicamente escolhidas pelo operador.

Dados os vetores de treinamento $\mathbf{x}_1^{(\ell)}, \ldots, \mathbf{x}_{N(\ell)}^{(\ell)}$ de tamanhos $N(\ell) \geq 2, \ell = 1, 2$, os estimadores de máxima verosimilhança para $\boldsymbol{\mu}^{(\ell)}$ e $\Sigma^{(\ell)}$ são dados por:

$$\widehat{\boldsymbol{\mu}^{(\ell)}} = \frac{1}{N(\ell)} \sum_{1 \leq i \leq N(\ell)} \mathbf{x}_i^{(\ell)}, \text{ e}$$

$$\widehat{\Sigma^{(\ell)}} = \frac{1}{N(\ell)} \sum_{1 \leq i \leq N(\ell)} \left(\mathbf{x}_i^{(\ell)} - \widehat{\boldsymbol{\mu}_i^{(\ell)}}\right)^{\mathrm{T}} \left(\mathbf{x}_i^{(\ell)} - \widehat{\boldsymbol{\mu}^{(\ell)}}\right)$$

Os valores $\widehat{\boldsymbol{\mu}^{(\ell)}}$ e $\widehat{\Sigma^{(\ell)}}$, $\ell = 1, 2$, são empregados no lugar de $\boldsymbol{\mu}^{(\ell)}$ e de $\Sigma^{(\ell)}$, $\ell = 1, 2$, respectivamente, na expressão (5). Diz-se então que "o algoritmo MV está treinado".

Os mesmos vetores $\mathbf{x}_1^{(\ell)}, \ldots, \mathbf{x}_{N(\ell)}^{(\ell)}$, $\ell = 1, 2$, são utilizados para treinar a rede neural. Esse processo de treinamento de redes neurais será discutido na próxima Seção. Diz-se, então, que o "algoritmo RN está treinado" e que "os algoritmos MV e RN têm o mesmo treinamento".

3 Redes Neurais

As redes neurais são implementações de um algoritmo baseado na pesquisa da neurociência sobre o funcionamento do cérebro. São estruturas de processamento, formadas pela interconexão de elementos chamados *neurônios*.

Cada neurônio tem um conjunto de entradas, um peso associado a cada uma delas e produz uma saída que é função dessas entradas e pesos.

O perceptron [19], um modelo típico de neurônio, pode ser visto na Figura 1.

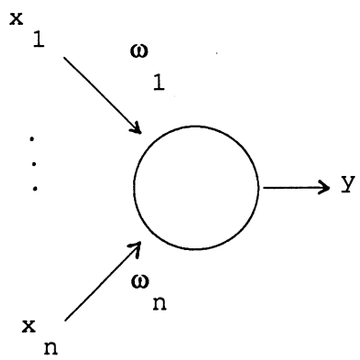

Figura 1: Esquema de neurônio: o perceptron.

Matematicamente tem-se, para cada neurônio, que se existem n entradas $(x_i)_{1\leq i\leq n}$ então existem n pesos $(w_i)_{1\leq i\leq n}$ associados às entradas. A soma ponderada das entradas é comparada a um certo valor chamado *limiar*. Se a soma é maior que o limiar então a saída é "1", e "0" caso contrário. Então, a saída é dada por:

$$y = \mathcal{F}\left(\sum_{1\leq i\leq n} x_i\, w_i - \theta\right), \tag{6}$$

onde $\mathcal{F}$ é a função dada por:

$$\mathcal{F}(x) = \begin{cases} 1 & \text{se } x > 0 \\ 0 & \text{caso contrário.} \end{cases} \tag{7}$$

Fica, assim, especificada a função de transferência do neurônio.

Podem-se considerar outras funções $\mathcal{F}$, dependendo da aplicação. Quando perceptrons são encadeados em camadas, é mais comum usar a função $\mathcal{F}$ sigmóide, dada por:

$$\mathcal{F}(x) = \left(1 + e^{-\beta x}\right)^{-1}, \tag{8}$$

onde β é o parâmetro que controla a *inclinação* da resposta.

Definida a função de transferência do nó, é necessário especificar um mecanismo de treinamento da rede, ou seja, um mecanismo de alteração dos pesos associados às entradas, de forma que a saída produzida se aproxime, até níveis desejados ou possíveis, da saída alvo, e assim a rede execute a tarefa desejada. Se esse mecanismo é guiado por um conjunto de exemplos onde se conhecem a entrada e a saída alvo para cada entrada, diz-se que é

um treinamento supervisionado. Diz-se que *a rede aprendeu* quando a saída produzida se aproxima satisfatoriamente da saída alvo e não há mais alterações significativas no valor dos pesos.

Uma regra de treinamento usada nas redes apresentadas acima é o algoritmo do treinamento do perceptron [10], dado, para cada neurônio simultaneamente por:

1. Definir $\omega_i(t)$, $0 \leq i \leq n$, o peso da entrada i no instante $t = 0$. Estes valores iniciais são tipicamente "pequenos" e "aleatórios". Atribuir $t = 0$, e definir θ.
   (a) Atribuir $t = t + 1$.
   (b) Apresentar a entrada $x_0(t), \ldots, x_n(t)$ e a saída desejada $d(t)$ para essa entrada.
   (c) Calcular $y(t) = \mathcal{F}\left(\sum_{i=1}^{n} \omega_i(t)x_i(t) - \theta\right)$.
   (d) Adaptar os pesos segundo a expressão

$$\omega_i(t+1) = \begin{cases} \omega_i(t) & \text{se } y(t) - d(t) = 0 \\ \omega_i(t) + x_i & \text{se } y(t) - d(t) = -1 \\ \omega_i(t) - x_i & \text{se } y(t) - d(t) = 1. \end{cases}$$

   (e) Voltar ao passo (1a) enquanto algum critério de parada não for satisfeito.
2. Finalizar com $(\omega_i)_{1 \leq i \leq n} = (\omega_i(t))_{1 \leq i \leq n}$, dizendo que a rede está treinada.

Existem variações quanto aos algoritmos de teinamento [2]. A prova de que se é possível classificar uma série de entradas então uma rede de perceptrons irá encontrá-la pode ser vista em [2].

Generalizando, considerem-se k neurônios. Para cada $1 \leq m \leq k$ neurônio tem-se $E(m)$ entradas e:

$$\sum_{1 \leq s \leq E(m)} e_s = E(m),$$

isto é, a cada uma das $(e_s)_{1 \leq s \leq \sum_{m=1}^{k} E(m)}$ entradas associa-se o peso ω_s. Ver Figura 2.

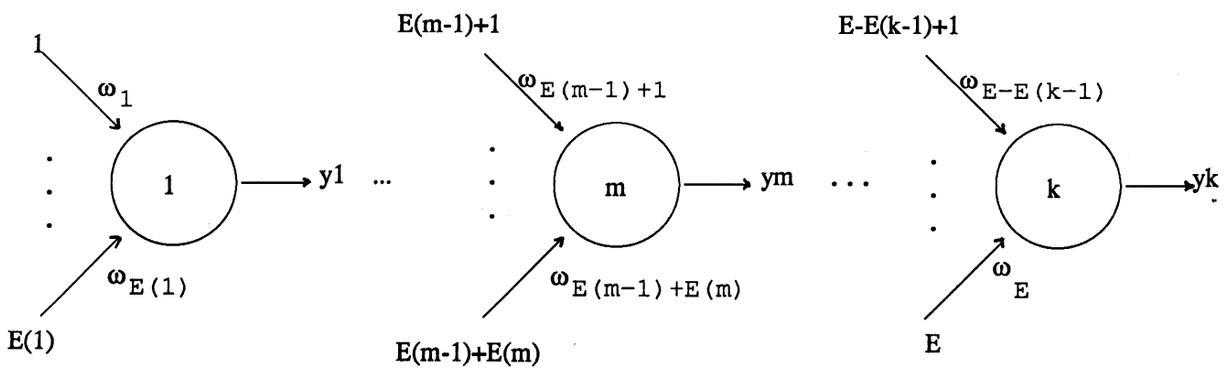

Figura 2: Esquema de rede neural.

A saída do neurônio m é dada por:

$$y_m = \mathcal{F}\left(\sum_{j=E(m-1)+1}^{E(m-1)+E(m)} x_j\omega_j - \theta\right)$$

onde $\mathcal{F}\colon \mathbb{R} \longrightarrow \mathbb{R}$ é a função de transferência do neurônio m. Tipicamente, consideram-se funções crescentes e $\mathcal{F}\colon \mathbb{R} \longrightarrow [0,1]$, como na equação (7) ou na equação (8).

A topologia da rede é dada pela associação dos dados a certas entradas $E(\cdot)$, e pela coleta de resultados em certas saídas $y_\cdot$. Para a compreensão da topologia, os neurônios são arranjados em O camadas onde:

- a entrada da camada $o = 1$, são os dados
- a saída da camada $o = O$, são os resultados
- a saída da camada $o \leq O - 1$ é a entrada da $o + 1$.

A escolha de uma topologia em particular, bem como da função de transferência, dependem de considerações a respeito da aplicação (volume dos dados, tempo disponível para efetuar o treinamento, hardware ou simulador a ser empregado, precisão dos resultados desejados, etc.).

4 A Rede Empregada

O problema sob estudo consiste se em classificar o vetor $\mathbf{x} = (x_1, x_2)^{\mathrm{T}} \in \mathbb{R}^2$ na classe $\ell = 1$ ou $\ell = 2$.

A topologia escolhida foi uma rede com 2 nós na entrada (um para cada componente de $\mathbf{x}$), 1 nó na saída (que fornece um valor aproximado para ℓ) e o menor número de camadas intermediárias, cada uma com o menor número de nós, de maneira a se obter resultados compatíveis com o método de máxima verosimilhança.

Tal escolha heurística da topologia visa obedecer o princípio de parssimoniosidade, isto é, deseja-se obter certos resultados com ferramentas o mais simples possíveis dentro de uma classe. A classe escolhida é a das redes com 2 nós na entrada, 1 nó na saída, $O \geq 2$ camadas com funções de transferência do tipo sigmóide. Finalmente, a rede escolhida tem $O = 3$ camadas com 1 nó na camada intermediária, vide Figura 3.

O algoritmo de treinamento utilizado é o chamado *regra delta generalizada*, ou *back-propagation* [2].

5 Comparação e Conclusões

Foram repetidas dez vezes as seguintes quinze situações: para cem amostras de classificação os algoritmos foram treinados com quarenta, vinte e dez amostras (isto é, $N(\ell) = 20, 10, 5$

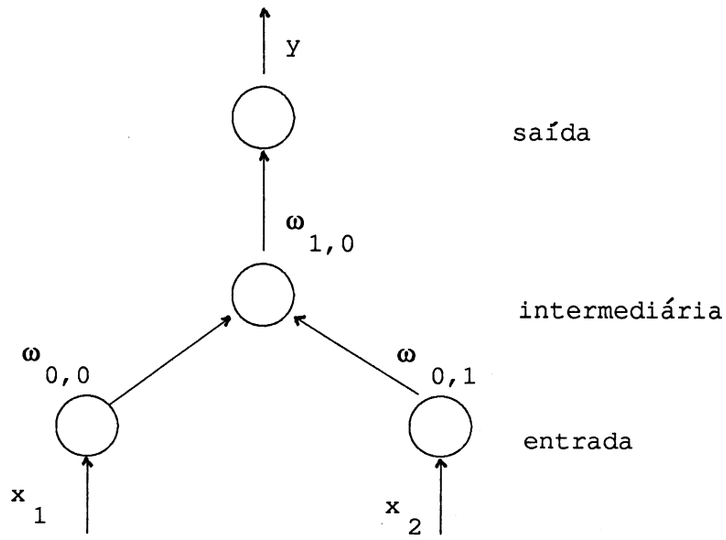

Figura 3: Rede neural empregada.

para $\ell = 1, 2$) para cada um dos seguintes casos,

$$\begin{array}{lll}
\text{Caso 1:} & \boldsymbol{\mu}^{(1)} = (0.7, 0.7)^{\mathrm{T}} & \Sigma^{(1)} = \begin{pmatrix} 0.05 & 0 \\ 0 & 0.05 \end{pmatrix}, \\
& \boldsymbol{\mu}^{(2)} = (-0.7, -0.7)^{\mathrm{T}} & \Sigma^{(2)} = \begin{pmatrix} 0.05 & 0 \\ 0 & 0.05 \end{pmatrix}; \\
\text{Caso 2:} & \boldsymbol{\mu}^{(1)} = (0.5, 0.5)^{\mathrm{T}} & \Sigma^{(1)} = \begin{pmatrix} 0.05 & -0.04 \\ -0.04 & 0.05 \end{pmatrix}, \\
& \boldsymbol{\mu}^{(2)} = (-0.5, -0.7)^{\mathrm{T}} & \Sigma^{(2)} = \begin{pmatrix} 0.05 & 0.03 \\ 0.03 & 0.05 \end{pmatrix}; \\
\text{Caso 3:} & \boldsymbol{\mu}^{(1)} = (0.5, 0.5)^{\mathrm{T}} & \Sigma^{(1)} = \begin{pmatrix} 0.05 & 0.04 \\ 0.04 & 0.05 \end{pmatrix}, \\
& \boldsymbol{\mu}^{(2)} = (-0.5, -0.7)^{\mathrm{T}} & \Sigma^{(2)} = \begin{pmatrix} 0.05 & 0.03 \\ 0.03 & 0.05 \end{pmatrix}; \\
\text{Caso 4:} & \boldsymbol{\mu}^{(1)} = (0.5, 0.5)^{\mathrm{T}} & \Sigma^{(1)} = \begin{pmatrix} 0.15 & 0.07348 \\ 0.07348 & 0.1 \end{pmatrix}, \\
& \boldsymbol{\mu}^{(2)} = (-0.5, -0.7)^{\mathrm{T}} & \Sigma^{(2)} = \begin{pmatrix} 0.9 & 0.2738 \\ 0.2738 & 0.17 \end{pmatrix}; \\
\text{Caso 5:} & \boldsymbol{\mu}^{(1)} = (0.3, 0.4)^{\mathrm{T}} & \Sigma^{(1)} = \begin{pmatrix} 0.2 & 0.0565 \\ 0.0565 & 0.1 \end{pmatrix}, \\
& \boldsymbol{\mu}^{(2)} = (-0.4, -0.3)^{\mathrm{T}} & \Sigma^{(2)} = \begin{pmatrix} 0.15 & -0.0492 \\ -0.0492 & 0.18 \end{pmatrix};
\end{array}$$

e para cada uma das dez vezes, foram registrados os números de erros na classificação das amostras de treinamento (matriz de confusão) e de amostras de classificação (matriz de desempenho).

Estas matrizes, que descrevem o desempenho dos algoritmos de classificação, estão definidas por:

$$\begin{pmatrix} 100 - F_{1,2} & F_{1,2} \\ F_{2,1} & 100 - F_{2,1} \end{pmatrix},$$

onde $F_{i,j}$ representa o número de amostras que, provindas (simuladas) da classe j foram atribuídas à classe i. Evidentemente, quanto menores forem os valores de $F_{i,j}$, tanto

para dados de treinamento como para dados de classificação, melhor será o desempenho do método. Dado que estão sendo comparados dois métodos, denotar-se-á $F^{\bullet}_{i,j}$, onde $\bullet \in \{\mathrm{MV}, \mathrm{RN}\}$, e são fornecidos, a seguir, as médias e os desvios padrões para as dez repetições das experiências.

Tabela 1: Matriz de confusão, treinamento com quarenta amostras: $N(\ell) = 20$, $\ell = 1, 2$.

Caso	$F^{\mathrm{MV}}_{1,2}$	$F^{\mathrm{MV}}_{2,1}$	$F^{\mathrm{RN}}_{1,2}$	$F^{\mathrm{RN}}_{2,1}$
1	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
2	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
3	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
4	1.50(1.18)	1.20(0.63)	1.30(1.25)	1.40(1.35)
5	1.60(0.70)	2.10(1.10)	1.60(0.70)	3.00(3.16)

Tabela 2: Matriz de desempenho, treinamento com quarenta amostras: $N(\ell) = 20$, $\ell = 1, 2$.

Caso	$F^{\mathrm{MV}}_{1,2}$	$F^{\mathrm{MV}}_{2,1}$	$F^{\mathrm{RN}}_{1,2}$	$F^{\mathrm{RN}}_{2,1}$
1	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
2	0.00(0.00)	0.10(0.32)	0.00(0.00)	0.00(0.00)
3	0.70(0.82)	1.40(0.84)	1.00(1.33)	1.40(0.84)
4	8.50(3.60)	8.20(4.40)	12.90(6.00)	7.20(3.73)
5	9.10(3.69)	12.10(3.60)	12.00(6.91)	12.40(5.38)

Das Tabelas 1 e 2 pode concluir-se que para o treinamento com quarenta amostras, $N(\ell) = 20$, $\ell = 1, 2$, os métodos MV e RN produzem, em média, resultados compatíveis para todos os casos considerados, tanto na classificação de amostras de treinamento como na classificação de amostras não empregadas no treinamento. Já os desviões padrões do método RN são maiores que os do método MV, apontando maior variabilidade no desempenho do método RN.

Das Tabelas 3 e 4 pode concluir-se que para o treinamento com vinte amostras $N(\ell) = 10$, $\ell = 1, 2$, e nos Casos 4 e 5, aparecem indícios de melhor desempenho do método RN com respeito ao método MV. Já nos Casos 1, 2 e 3, a diferença (que aponta no mesmo sentido) é pouco significativa.

Das Tabelas 5 e 6 surge evidência clara do melhor desempenho do método RN com respeito ao método MV, quando o tamanho da amostra de treinamento é pequeno e as classes são pouco separáveis (Caso 5): enquanto as matrizes de confusão são compatíveis, os valores da matriz de desempenho, para este caso, são expressivamente melhores para o método RN.

Os resultados desta experiência, para o desempenho do método MV, podem ser expli-

Tabela 3: Matriz de confusão, treinamento com vinte amostras: $N(\ell) = 10$, $\ell = 1, 2$.

Caso	$F_{1,2}^{\mathrm{MV}}$	$F_{2,1}^{\mathrm{MV}}$	$F_{1,2}^{\mathrm{RN}}$	$F_{2,1}^{\mathrm{RN}}$
1	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
2	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
3	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
4	1.00(0.67)	0.20(0.42)	0.40(0.51)	1.10(0.99)
5	0.60(0.84)	1.40(2.11)	1.20(0.91)	1.00(1.24)

Tabela 4: Matriz de desempenho, treinamento com vinte amostras: $N(\ell) = 10$, $\ell = 1, 2$.

Caso	$F_{1,2}^{\mathrm{MV}}$	$F_{2,1}^{\mathrm{MV}}$	$F_{1,2}^{\mathrm{RN}}$	$F_{2,1}^{\mathrm{RN}}$
1	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
2	0.10(0.32)	0.10(0.32)	0.00(0.00)	0.00(0.00)
3	3.40(5.25)	1.20(1.69)	0.40(0.70)	3.67(6.24)
4	8.00(6.48)	11.80(6.25)	8.80(5.02)	15.50(7.35)
5	10.00(3.43)	12.90(3.10)	2.70(5.27)	12.90(4.38)

Tabela 5: Matriz de confusão, treinamento com dez amostras: $N(\ell) = 5$, $\ell = 1, 2$.

Caso	$F_{1,2}^{\mathrm{MV}}$	$F_{2,1}^{\mathrm{MV}}$	$F_{1,2}^{\mathrm{RN}}$	$F_{2,1}^{\mathrm{RN}}$
1	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
2	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
3	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
4	0.30(0.48)	0.00(0.00)	0.30(0.41)	0.10(0.32)
5	0.45(0.52)	0.70(0.67)	0.40(0.52)	0.40(0.51)

Tabela 6: Matriz de desempenho, treinamento com dez amostras: $N(\ell) = 5$, $\ell = 1, 2$.

Caso	$F_{1,2}^{\mathrm{MV}}$	$F_{2,1}^{\mathrm{MV}}$	$F_{1,2}^{\mathrm{RN}}$	$F_{2,1}^{\mathrm{RN}}$
1	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
2	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
3	5.80(7.40)	1.80(2.44)	1.40(1.17)	1.00(0.33)
4	7.50(6.74)	22.40(15.65)	16.90(8.87)	11.20(10.56)
5	20.80(14.87)	20.80(19.84)	20.10(14.45)	8.50(6.55)

cados pelo fato da qualidade da estimação pelo método de máxima verossimilhança ser fortemente dependente do tamanho da amostra [4, 12].

6 Informação Computacional

O problema de geração de números pseudoaleatórios por computador é quase tão antigo quanto os computadores digitais. Parte do interesse que o assunto tem despertado na comunidade da Ciência da Computação deve-se, provavelmente, ao fato da geração destes números ir de encontro ao paradigma de máquina de calcular: deseja-se ter comportamento aleatório de uma engenhoca que *por definição* é determinística. Outra motivação, não menos intensa do que a comentada, é que muitas áreas requerem o uso de números que apresentam comportamento similar ao de ocorrências de variáveis aleatórias. Para uma discusão sobre estas e outras motivações, o leitor pode consultar o texto [7] e as referências aí indicadas.

Para as implementações apresentadas neste trabalho foi obtida a biblioteca de sub-rotinas `RANLIB` [5, 17] através do nodo de distribuição `STATLIB` (disponível no endereço eletrônico `statlib@lib.stat.cmu.edu` —Internet—). Esta biblioteca oferece um leque bastante largo de rotinas (nas linguagens de programação `FORTRAN` e `C`), em forma gratuita e com a única condição de se citar a origem dos algoritmos empregados.

O algoritmo mais empregado, neste trabalho, foi o de geração de ocorrências de variáveis aleatórias normails k-variadas (`genmn` [1]).

Para a realização da experiência aqui comentada foram implementados dois programas: um para gerar as ocorrências das variáveis aleatórias $\mathcal{N}2^{(\ell)}$, pelo uso de [5], e o outro para se fazer a classificação pelo método de RN. Esta classificação é efetuada pelo uso de um programa, na linguagem de programação `C`, que simula a rede mencionada na Seção 4 em uma estação de trabalho. Para se fazer a classificação pelo método de MV, foi empregado o pacote `MatLab` V. 4.0a.

7 Extensões

A principal limitação encontrada foi a dificuldade de *comunicação* entre as ferramentas mencionadas (os programas de simulação, de RN e o pacote `MatLab`). Devido a esta limitação, o número de replicações efetuadas para cada modelo foi fixada em "10", um número bastante modesto para uma experiência Monte Carlo.

Já está sendo montado um ambiente unificado, capaz de realizar as três tarefas, que permitirá realizar experiências deste tipo, porém, de maior porte. Uma vez disponível esse ambiente, serão considerados outros casos de interesse para a comparação de desempenho dos algoritmos, por exemplo:

1. situações onde amostras de treinamento e/ou de classificação provêm da distribuição $\mathcal{N}k$, mas apresentam *contaminação* [6, 16];
2. situações onde amostras de treinamento e/ou de classificação provêm de distribuições diferentes da $\mathcal{N}k$: dupla Exponencial, Cauchy, Gamma, etc., mas se

mantém o algoritmo MV tal como aqui mencionado;

3. problemas de maior dimensionalidade [16], isto é, $k \geq 2$,

entre outros.

Desta maneira, e se dispondo de informação a respeito dos tempos de treinamento e de classificação para ambos métodos, poderão tirar-se conclusões a respeito dos seus desempenhos para a tarefa aqui estudada.

8 Agradecimentos

Este trabalho foi parcialmente desenvolvido com recursos do Projeto Temático de Equipe No. 91/3532-2 da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP).

Bibliografia

[1] Ahrens, J.H.; Dieter, U. Extensions of Forsythe's method for random sampling from the normal distribution. *Mathematics of Computation*, 27(124):927–937, Oct. 1973.

[2] Beale, R.; Jackson, T *Neural computing: an introduction.* USA, IOP Publishing, 1992.

[3] Benediktsson, J.A.; Swain, P.H.; Ersoy, O.K. Neural Network Aproaches Versus Statistical Methods in Classification of Multisource Remote Sensing Data. *IEEE Transactions on Geoscience and Remote Sensing*, 7(28):5540–552, Jul. 1990.

[4] Bickel, P.J.; Doksum, K.A. *Mathematical statistics: basic ideas and selectd topics.* USA, Holden-Day, Inc., 1977. 493 p.

[5] Brown, B.W.; Lovato, J. *RANLIB.C: library of C routines for random number generation.* Houston, TX, s.ed., s.d.

[6] Bustos, O.H. Outliers e robustez. *Revista Brasileira de Estatística*, 49(191):5–25, jan.–jun. 1988.

[7] Bustos, O.H.; Frery, A.C. *Simulação estocástica: teoria e algoritmos (versão completa).* Rio de Janeiro, Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq)/Instituto de Matemática Pura e Aplicada, 1992. 148 p. (Monografias de Matemática, 49).

[8] Câmara, G.; Souza, R.C.M.; Freitas, U.M.; Casanova, M.A. SPRING: processamento de imagens e dados georeferenciados. In: Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens, 5., Águas de Lindóia, nov. 1992. São José dos Campos, INPE/SBC, 1992, p. 233–242.

[9] Crósta, A.P. *Processamento digital de imagens de sensoriamento remoto.* Campinas, UNICAMP, 1992.

[10] Duda, R.; Hart, E. *Pattern classification and scene analysis.* USA, John Wiley & Sons, 1973.

[11] Dutra, L.V.; Frery, A.C.; Krug, T.; Mascarenhas, N.D.A.; Sant'Anna, S.J.S.; Yanasse, C.C.F. *Alguns aspectos de modelagem estatística de dados de sensoriamento remoto.* Curitiba, INPE/SBC, maio 1993. Notas de curso do Simpósio Brasileiro de Sensoriamento Remoto, 7.

[12] James, B.R. *Probabilidade: um curso em nível intermediário.* Rio de Janeiro, Instituto de Matemática Pura e Aplicada - CNPq, 1981. 304 p. (Projeto Euclides).

[13] Hammerstron, D. Neural networks at work. *IEEE Spectrum*, 30(6):26–32, Jun. 1993.

[14] Hammerstron, D. Working with neural networks. *IEEE Spectrum*, 30(7):46–53, Jul. 1993.

[15] Instituto Nacional de Pesquisas Espaciais. *Sistema de tratamento de imagens (SITIM): manual do usuário.* São José dos Campos, INPE, sd.

[16] Johnson, M.E. *Multivariate statistical simulation.* New York, John Wiley & Sons, 1987. 230 p.

[17] L'Ecuyer, P.; Côte, S. Implementing a random number package with splitting facilities. *ACM Transactions on Mathematical Software*, 17:98-111, 1991.

[18] Lewis, P.A.; Orav, E.J. *Simulation methodology for statisticians, operation analysts and engineers.* Cole Advanced Books & Software, Wadsworth & Books, Pacific Grove, California, 1989. v. 1.

[19] Rosemblat, F. The perceptron: a probabilstic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958