SY19
Réseaux de neurones: optimisation de l'architecture

T. Denœux

1 Introduction

Afin d'utiliser le modèle du PMC, il convient de déterminer le nombre de neurones cachés, c'est-à-dire la complexité du réseau. En pratique, augmenter le nombre de neurones cachés permet de réduire le risque empirique, mais il n'en est pas de même pour le risque théorique. Nous allons expliquer ce phénomène et présenter des méthodes pratiques pour déterminer la complexité optimale du réseau.

2 Contrôle de la complexité

Soit $\mathcal{G}$ l'ensemble des fonctions de prédiction calculable par un PMC avec une architecture donnée. La meilleure fonction de décision possible $g^*_{\mathcal{G}}$ dans $\mathcal{G}$ est celle qui minimise le risque, c'est-à-dire :

$$g^*_{\mathcal{G}} = \arg\min_{g \in \mathcal{G}} R(g)$$

avec

$$R(g) = \mathbb{E}_{\mathbf{X},Y}\left[L(g(\mathbf{X}), Y)\right],$$

où L est la fonction de coût.

Par comparaison, la fonction de décision optimale g^* est définie par

$$g^* = \arg\min_{g} R(g).$$

Etant donné que $g^*_{\mathcal{G}}$ est une solution plus contrainte que g^*, $R(g^*_{\mathcal{G}})$ est en général plus grand que $R(g^*)$. La différence $R(g^*_{\mathcal{G}}) - R(g^*)$ est appelée *erreur d'approximation*. Afin de réduire cette erreur, il semble donc que l'on ait intérêt à choisir $\mathcal{G}$ le plus grand possible, c'est-à-dire autant de neurones cachés que possible.

Cependant, on ne minimise pas réellement le risque $R(g)$, mais son estimation sur l'ensemble d'apprentissage, c'est-à-dire le risque empirique :

$$\widehat{R}(g) = \frac{1}{n}\sum_{i=1}^{n} L(g(\mathbf{x}_i), y_i). \tag{1}$$

L'apprentissage a pour but de trouver :

$$\widehat{g}_{\mathcal{G}} = \arg\min_{g \in \mathcal{G}} \widehat{R}(g).$$

Il est clair que

$$R(\widehat{g}_{\mathcal{G}}) \geq R(g^*_{\mathcal{G}}) \geq R(g^*).$$

La différence $R(\widehat{g}_{\mathcal{G}}) - R(g^*_{\mathcal{G}})$ est appelée *erreur d'estimation*. Quand on augmente la taille de $\mathcal{G}$ (la complexité), l'erreur d'estimation a tendance à augmenter, car $\widehat{g}_{\mathcal{G}}$ peut devenir très différent de $g^*_{\mathcal{G}}$.

Les erreurs d'approximation et d'estimation varient donc en sens inverse en fonction de la taille de $\mathcal{G}$. Pour minimiser la somme des deux, il faut donc trouver un compromis. C'est le problème de la *sélection de modèle*.

Pour cela, on définit en pratique une suite de structures emboîtées de complexité croissante, dépendant d'un hyperparamètre λ :

$$\mathcal{G}_1 \subset \mathcal{G}_2 \subset \ldots \subset \mathcal{G}_\lambda \subset \ldots$$

Pour chaque valeur de λ, on estime $R(\widehat{g}_{\mathcal{G}_\lambda})$, à l'aide d'un ensemble de validation ou par validation croisée. On retient finalement la valeur λ^* correspondant au risque estimé le plus faible. Pour mettre en œuvre cette approche, il faut définir :

- Une méthode d'estimation du risque $R(\widehat{g}_{\mathcal{G}_\lambda})$;
- Une famille de modèles $(\mathcal{G}_\lambda)$.

Ces deux problèmes sont traités ci-dessous.

3 Estimation du risque

3.1 Méthode de resubstitution

La méthode de resubstitution consiste à estimer le risque $R(g)$ d'une fonction de décision g par le risque empirique (1). Cette méthode ne convient pas pour la sélection de modèle, car elle conduit toujours à choisir le modèle le plus complexe, qui ne correspond pas généralement au minimum du risque théorique.

3.2 Méthode de l'ensemble de validation

Cette méthode consiste à partitionner aléatoirement les données initiales $\mathcal{D}$ en un ensemble d'apprentissage $\mathcal{L}$ et un *ensemble de validation* $\mathcal{V}$, avec $\mathcal{D} = \mathcal{L} \cup \mathcal{V}$ et $\mathcal{L} \cap \mathcal{V} = \emptyset$. On prend en général $\text{card}(\mathcal{L}) > \text{card}(\mathcal{V})$ car l'apprentissage de la règle de décision est un problème plus difficile que l'estimation du risque. Typiquement, on réserve un tiers ou un quart des données initiales pour l'estimation du risque.

Soit $\mathcal{V} = \{(\mathbf{x}'_1, y'_1), \ldots, (\mathbf{x}'_{n'}, y'_{n'})\}$. Ayant construit la fonction de décision g à l'aide de l'ensemble $\mathcal{L}$, on pose :

$$\widehat{R}_v(g) = \frac{1}{n'} \sum_{i=1}^{n'} L\left(g(\mathbf{x}'_i), y'_i\right).$$

Cette méthode est rigoureuse car elle fournit un estimateur sans biais du risque. En revanche, elle nécessite un ensemble de données de taille importante, les données de validation ne pouvant être utilisées pour l'apprentissage.

3.3 Méthode de validation croisée

Dans cette méthode, on cherche à estimer l'espérance R_n du risque d'une fonction de prédiction basé sur un modèle $\mathcal{G}$ donné, construit à l'aide d'un ensemble de données de taille n issu de la même distribution que $\mathcal{D}$:

$$R_n = \mathbb{E}_{\mathcal{D}_n} R(\widehat{g}_{\mathcal{D}_n}),$$

$\mathcal{D}_n$ étant un échantillon aléatoire de taille n, et $\widehat{g}_{\mathcal{D}_n}$ une fonction de prédiction construite à partir de $\mathcal{D}_n$.

Pour estimer R_n, on construit successivement plusieurs fonctions de décision à partir de plusieurs ensembles d'apprentissage : on parle de *méthode de rééchantillonnage*. Les différents ensembles d'apprentissage sont construits de la manière suivante.

On partitionne les données initiales $\mathcal{D}$ en V blocs $\mathcal{D}_{(1)}, \ldots, \mathcal{D}_{(V)}$ de tailles à peu près égales. Soit $n_{(k)} = \operatorname{card} \mathcal{D}_{(k)}$.

Soit $\widehat{g}^{-k}$ la fonction de décision construite à partir de l'ensemble d'apprentissage $\mathcal{D} \setminus \mathcal{D}_{(k)}$, et $\widehat{R}_{(k)}$ son risque estimé sur $\mathcal{D}_{(k)}$:

$$\widehat{R}_{(k)} = \frac{1}{n_{(k)}} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}_{(k)}} L\left(\widehat{g}^{-k}(\mathbf{x}_i), y_i\right).$$

L'estimateur de validation croisée de R_n est défini par

$$\widehat{R}_{CV} = \frac{1}{n} \sum_{k=1}^{V} n_{(k)} \widehat{R}_{(k)}.$$

On choisit typiquement V de l'ordre de 5 à 10. Dans le cas $V = n$, on parle de méthode *leave-one-out*.

Remarquons que $\widehat{r}_{CV}$ est un estimateur biaisé (pessimiste) qui tend à surestimer r_n, car il est obtenu en moyennant les risques estimés de V fonctions de décision construites à partir d'ensembles d'apprentissage de taille $n(1 - 1/V)$. Le biais est d'autant plus faible que V est grand.

Cette méthode permet une utilisation « optimale » des données (puisque chaque exemple est utilisé tour à tour pour l'apprentissage et pour le test), au prix d'un temps de calcul plus important (il faut apprendre V classifieurs différents).

3.4 Méthode du bootstrap

Le bootstrap est une méthode générale d'estimation de la loi de probabilité d'une statistique $T(X_1, \ldots, X_n)$.

Principe général

La méthode consiste à constituer B échantillons de bootstrap à partir de $\mathcal{D}$. Un échantillon de bootstrap $\mathcal{D}_b^*$ a la même taille que l'échantillon initial. Il s'obtient en faisant n tirages *avec remise* dans $\mathcal{D}$. Il peut donc contenir plusieurs fois la même observation.

Soit T_b^* la statistique calculée à partir de l'échantillon de bootstrap $\mathcal{D}_b^*$. Pour estimer une caractéristique de la distribution de T, il suffit de calculer la caractéristique correspondante pour la distribution empirique $T_1^*, \ldots, T_b^*$.

Par exemple, on estimera la variance σ^2 de T par

$$\widehat{\sigma}^2_{boot} = \frac{1}{B}\sum_{b=1}^{B}\left[T^*_b - \left(\frac{1}{B}\sum_{b=1}^{B} T^*_b\right)\right]^2$$

Application à l'estimation du risque

Comme dans la méthode de validation croisée, on cherche à estimer le risque moyen R_n d'une fonction de décision construite à partir d'un ensemble d'apprentissage aléatoire de taille n.

Soient $\mathcal{D}^*_1, \ldots, \mathcal{D}^*_B$ B échantillons de bootstrap (ce sont des ensembles d'apprentissage de même taille que $\mathcal{D}$, soit n). Soit $\widehat{g}^*_b$ la fonction de décision construite à partir de $\mathcal{D}^*_b$. Son risque estimé sur l'ensemble de données initial $\mathcal{D}$ est :

$$\widehat{R}^*_b = \frac{1}{n}\sum_{i=1}^{n} L(\widehat{g}^*_b(\mathbf{x}_i), y_i).$$

Une première application de la méthode du bootstrap consiste à estimer le risque par :

$$\widehat{R}_{boot} = \frac{1}{B}\sum_{b=1}^{B}\widehat{R}^*_b = \frac{1}{nB}\sum_{i=1}^{n}\sum_{b=1}^{B} L(\widehat{g}^*_b(\mathbf{x}_i), y_i).$$

Le problème est ici que l'estimateur $\widehat{R}_{boot}$ est optimiste, car $\mathcal{D}^b \cap \mathcal{D} \neq \emptyset$: certains exemples sont utilisés à la fois pour l'apprentissage et pour le test.

Une solution à ce problème consiste à évaluer chaque fonction de décision $\widehat{g}^*_b$ en utilisant uniquement les exemples $(\mathbf{x}_i, y_i)$ qui n'appartiennent pas à $\mathcal{D}^*_b$. Soit $\mathcal{B}^{-i}$ l'ensemble des indices b des échantillons de bootstrap qui ne contiennent pas l'exemple i. On pose

$$\widehat{R}^{(1)}_{boot} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|\mathcal{B}^{-i}|}\sum_{b\in\mathcal{B}^{-i}} L(\widehat{g}^*_b(\mathbf{x}_i), y_i).$$

Cette méthode est meilleure que la précédente, mais elle présente encore un inconvénient : dans chaque échantillon de bootstrap $\mathcal{D}^*_b$, certains $\mathbf{x}_i$ apparaissent plusieurs fois. Le nombre d'exemples *distincts* dans chaque échantillon de bootstrap est donc en général strictement inférieur à n : tout se passe comme si ces échantillons étaient en réalité plus petits que $\mathcal{D}$.

La probabilité qu'un exemple i n'appartienne pas à $\mathcal{D}^*_b$ est

$$\mathbb{P}(\mathbf{x}_i \notin \mathcal{D}^*_b) = \left(1 - \frac{1}{n}\right)^n \approx 1/e.$$

La probabilité qu'un exemple i appartienne à $\mathcal{D}^*_b$ est donc :

$$\mathbb{P}(\mathbf{x}_i \in \mathcal{D}^*_b) \approx 1 - 1/e \approx 0.632.$$

On en déduit que le nombre moyen d'exemples distincts dans chaque échantillon de bootstrap est environ $0.632n$.

Ces considérations ont conduit à définir empiriquement l'estimateur 0.632 comme une moyenne pondérée de l'estimateur de resubstitution $\widehat{R}_R$ (optimiste) et de l'estimateur $\widehat{R}^{(1)}_{boot}$ (pessimiste) :

$$\widehat{R}^{(0.632)}_{boot} = 0.368\,\widehat{R}_R + 0.632\,\widehat{R}^{(1)}_{boot}.$$

4 Choix d'une famille de modèles

4.1 Essai de plusieurs architectures

C'est l'approche la plus simple. Elle consiste à faire varier le nombre de neurones cachés (voire le nombre de couches cachées) entre une valeur minimale et une valeur maximale. Pour chaque architecture, on estime le risque par une des méthodes ci-dessus, et on choisit l'architecture correspondant au risque estimé minimal. Une fois choisie l'architecture, on refait généralement un approentissage avec l'ensemble des données.

4.2 Régularisation

Plutôt que de faire varier le nombre de neurones cachés, on peut fixer ce nombre à une valeur assez grande et rechercher une solution suffisamment « régulière » en ajoutant au critère d'erreur $\widehat{R}$ un terme de pénalisation P :

$$C = \widehat{R} + \lambda P,$$

λ étant un *hyperparamètre*. Le rôle de P est de pénaliser les fonction de décision trop complexes. Par exemple, on peut poser :

$$P = \sum_j w_j^2$$

ou

$$P = \sum_j \frac{w_j^2}{1 + w_j^2}.$$

Ces deux fonctions de pénalisation sont diminuées lorsque certains poids w_j sont rendus nuls ou très faibles. L'hyperparamètre λ permet donc de régler la complexité de la solution.

4.3 Elagage

Après avoir obtenu un minimum du risque empirique pour une architecture donnée, cette méthode consiste à élaguer le réseau en supprimant des poids ou des unités cachées. En itérant le processus, on définit ainsi une suite de modèles de complexités décroissantes.

Par exemple, dans la méthode OBD *(Optimal Brain Damage)*, on évalue l'impact de la suppression d'une connexion sur la fonction d'erreur après convergence en calculant les dérivées secondes de l'erreur $\widehat{R}(\mathbf{w})$ par rapport aux poids.

En faisant un développement limité de l'erreur au voisinage du minimum $\mathbf{w}^*$, on obtient :

$$\widehat{R}(\mathbf{w}) = \widehat{R}(\mathbf{w}^*) + (\mathbf{w} - \mathbf{w}^*)' \frac{d\widehat{R}}{d\mathbf{w}}(\mathbf{w}^*) + \frac{1}{2} (\mathbf{w} - \mathbf{w}^*)' H(\mathbf{w}^*) (\mathbf{w} - \mathbf{w}^*) + \epsilon,$$

où $H(\mathbf{w}^*)$ est la matrice hessienne en $\mathbf{w}^*$. Le gradient s'annulant en $\mathbf{w}^*$, il vient :

$$\Delta\widehat{R}(\mathbf{w}) = \widehat{R}(\mathbf{w}) - \widehat{R}(\mathbf{w}^*) \approx \frac{1}{2} (\mathbf{w} - \mathbf{w}^*)' H(\mathbf{w}^*) (\mathbf{w} - \mathbf{w}^*).$$

En supposant la matrice hessienne diagonale :

$$\Delta\widehat{R}(\mathbf{w}) \approx \frac{1}{2}\sum_j \left(w_j - w_j^*\right)^2 h_{jj} = \sum_j \delta w_j^2 \frac{\partial^2 \widehat{R}}{\partial w_j^2}.$$

Si on met à zéro le poids w_j en laissant les autres poids inchangés, on a $\delta w_i = -w_j^*$ et $\delta w_k = 0$ pour $k \neq j$. L'accroissement consécutif de l'erreur est alors

$$\Delta\widehat{R} \approx \frac{1}{2}(w_j^*)^2 \frac{\partial^2 \widehat{R}}{\partial w_j^2}.$$

On a donc intérêt à annuler les poids w_j pour lesquels la quantité $(w_j^*)^2 \frac{\partial^2 \widehat{R}}{\partial w_j^2}$ est la plus faible. En pratique, on fixe la proportion de connexions à élaguer à chaque itération de la procédure d'élagage.