Systèmes d'inférence floue auto-évolutifs : apprentissage incrémental pour la reconnaissance de gestes manuscrits

Abdullah Almousa Almaksour, Eric Anquetil

Systèmes d'inférence floue auto-évolutifs : apprentissage incrémental pour la reconnaissance de gestes manuscrits

Abdullah Almaksour, Eric Anquetil

INSA de Rennes, Avenue des Buttes de Coesmes, F-35043 Rennes
CNRS, UMR IRISA, Campus de Beaulieu, F-35042 Rennes
Université Européenne de Bretagne, France
{Abdullah.Almaksour, Eric.Anquetil}@irisa.fr

RÉSUMÉ. *Nous présentons dans ce papier une nouvelle méthode pour la conception de moteurs de reconnaissance de gestes manuscrits personnalisables et auto-évolutifs, c'est-à-dire capables de s'adapter au style d'écriture et aux habitudes de chacun, sans toutefois nécessiter de période d'apprentissage fastidieuse. Nous utilisons une approche d'apprentissage incrémental de classifieurs basés sur les systèmes d'inférence floue de type Takagi-Sugeno. Cette approche comprend d'une part, une adaptation des paramètres linéaires associés aux conclusions des règles en utilisant la méthode des moindres carrés récursifs, et d'autre part, un apprentissage incrémental des prémisses de ces règles afin de modifier les fonctions d'appartenance suivant l'évolution de la densité des données dans l'espace de classification.*

ABSTRACT. *We present in this paper a new method for the design of customizable self-evolving handwriting recognition systems, which are able to adapt to writing style and needs of each writer, without require time-consuming re-learning process. The presented approach is based on first-order Takagi-Sugeno fuzzy inference system. This approach involves first an incremental clustering and adaptation of the premise part of the system, and secondly, an incremental learning of the linear consequents parameters of the system using a modified version of the Recursive Least Square method.*

MOTS-CLÉS : *Apprentissage incrémental, système d'inférence floue, reconnaissance de tracés manuscrits*

KEYWORDS: *Incremental learning, fuzzy inference system, handwriting recognition.*

1. Introduction

Avec l'émergence des assistants personnels numériques (PDA) et des téléphones mobiles de nouvelle génération (smartphones) utilisant des interfaces orientées stylo, les performances des systèmes de reconnaissance de l'écriture manuscrite en-ligne doivent être optimales. De plus en plus d'efforts sont nécessaires pour rendre ces systèmes plus robustes et adaptables afin de répondre aux nouveaux besoins des utilisateurs. Parmi ces nouveaux besoins, il apparaît intéressant de pouvoir proposer à l'utilisateur de choisir son propre groupe de gestes et de les assigner à différentes commandes interactives, par exemple, « copier », « coller », « annuler », etc. Ce contexte applicatif impose des contraintes spécifiques à la technique d'apprentissage utilisée. L'objectif est de rendre le système capable d'apprendre rapidement en utilisant peu de données. En effet, les utilisateurs sont rarement prêts à saisir chaque nouveau symbole plus d'une douzaine de fois pour l'apprentissage du système. Pour répondre à ces besoins, cette étude propose un algorithme original d'apprentissage incrémental pour les systèmes de reconnaissance de tracés manuscrits en-ligne.

La difficulté principale de cette approche est de construire « à la volée » un classifieur de tracés manuscrits, en s'appuyant sur très peu de connaissances disponibles (quelques exemples d'apprentissage), puis de l'adapter progressivement afin d'atteindre un fort taux de reconnaissance le plus rapidement possible.

Un algorithme d'apprentissage incrémental est défini dans (Polikar *et al.*, 2001) comme répondant aux critères suivants : il doit être capable d'apprendre des connaissances supplémentaires à partir des nouvelles données ; il ne doit pas nécessiter l'accès aux données d'origine (c'est-à-dire les données qui ont été utilisées pour apprendre le classifieur actuel) ; il doit préserver les connaissances déjà acquises ; et il doit être en mesure d'apprendre de nouvelles classes susceptibles d'être introduites avec de nouvelles données. Ces quatre points, qui s'appliquent pour tout problème général d'apprentissage incrémental, correspondent parfaitement aux caractéristiques particulières de notre problème, un algorithme d'apprentissage incrémental pour un système de reconnaissance auto-évolutif.

Il existe plusieurs algorithmes proposés pour effectuer l'apprentissage incrémental, bien que beaucoup d'entre eux ne répondent pas à tous les critères pour être considérés comme des approches d'apprentissage incrémental.

Nous pouvons distinguer principalement deux types d'algorithmes d'apprentissage incrémental : les algorithmes d'apprentissage incrémental de paramètres et les algorithmes d'apprentissage incrémental de structure et de paramètres. L'apprentissage incrémental de paramètres peut être assimilé à ce que l'on appelle « adaptation ». Dans ces systèmes, la structure est fixe et initialisée au départ alors que les paramètres du système sont appris progressivement en fonction des données d'apprentissage disponibles. Quelques exemples de ces systèmes sont présentés dans (Jr. *et al.*, 2007), (Jang, 1993) et (Mouchere *et al.*, 2007).

La plupart des algorithmes d'apprentissage incrémental de structure et de paramètres sont basés sur le principe de l'algorithme de clustering ART (Carpenter *et al.*, 1988), tel que (Carpenter *et al.*, 1992), (Sadri *et al.*, 2006), (Gary G. Yen, 2001) et (Lughofer, 2008). Le problème principal de ces systèmes est qu'ils sont sensibles à la sélection du paramètre de vigilance (qui correspond au seuil de distance qui contrôle la création d'un nouveau cluster), aux niveaux de bruit dans les données d'apprentissage et à l'ordre dans lequel les données d'apprentissage sont présentées.

Une autre approche que l'on rencontre souvent en apprentissage incrémental s'appuie sur des systèmes à base d'ensemble de classifieurs, comme dans (Polikar *et al.*, 2001) et (Minku *et al.*, 2009). Dans ces systèmes, au lieu de créer de nouveaux clusters pour modéliser les nouvelles connaissances, se sont de nouveaux classifieurs qui sont appris et ajoutés au système de façon incrémentale après l'acquisition d'une certaine quantité de données. La performance du système est donc basée sur la performance synergique d'un ensemble de classifieurs faibles. Cette technique peut être considérée comme un processus d'apprentissage incrémental hors-ligne (c'est-à-dire non-instantané). Elle est souvent utilisée pour apprendre incrémentalement une base de taille importante en opérant une séquence d'apprentissage « batch » sur des sous-ensembles de cette base. Cependant, les classifieurs créés sont très difficilement adaptables après leur apprentissage.

Dans un algorithme d'apprentissage incrémental, la base d'apprentissage n'est pas disponible *a priori*, puisque les exemples d'apprentissage arrivent au fil du temps. Bien que les systèmes d'apprentissage en-ligne mettent à jour et améliorent constamment leurs modèles, ils ne sont pas forcement tous basés sur une approche incrémentale. Dans certains systèmes, le modèle est complètement reconstruit à chaque étape d'apprentissage incrémental en utilisant l'ensemble des données disponibles, ce sont des systèmes avec mémoire d'exemples complète (full instance memory) (Reinke, 1988). Par contre, si l'algorithme d'apprentissage modifie le modèle uniquement en fonction du dernier exemple d'apprentissage, c'est un algorithme incrémental sans mémoire d'exemples (no instance memory) (Littlestone *et al.*, 1994) (Littlestone, 1991). Une troisième approche est celle des systèmes à mémoire partielle d'exemples, qui sélectionnent et gardent un sous-ensemble réduit d'exemples pour les utiliser dans les prochaines étapes d'apprentissage (Aha *et al.*, 1991).

Dans ces travaux, on constate que les modèles à base de prototypes sont généralement les mieux adaptés aux problèmes d'apprentissage incrémental rapide, où le modèle doit être modifié pour chaque nouvel exemple sans disposer d'une mémoire complète des exemples précédents.

Dans ce papier, nous proposons un nouvel algorithme d'apprentissage incrémental de structure et de paramètres avec une mémoire partielle d'exemples. Ce système est utilisé dans notre contexte pour la reconnaissance de gestes manuscrits en-ligne, et il est capable d'apprendre une nouvelle classe tout en continuant à évoluer, exemple après exemple, sans utiliser de données antérieures.

Le reste de cet article est organisé comme suit. Dans la section 2, nous présentons nos travaux antérieurs et d'autres travaux connexes qui ont inspiré le modèle proposé dans ce document. Ensuite, nous décrivons dans la section 3 l'architecture du système utilisé. Les différents éléments de notre algorithme d'apprentissage sont détaillés dans la section 4. La section 5 étudie les résultats des évaluations expérimentales.

2. Travaux connexes

Nous avons présenté dans les travaux antérieurs (Almaksour *et al.*, 2008) (Almaksour *et al.*, 2009) un modèle d'apprentissage incrémental basé sur un Système d'Inférence Floue (SIF). Nous avons choisi ce système de classification à base de prototypes pour sa nature flexible qui le rend capable de s'adapter et d'évoluer selon les nouvelles données. Le SIF utilisé est du type Takagi-Sugeno d'ordre zero (Takagi *et al.*, 1985). Les règles floues font le lien entre les modèles intrinsèques (prémisses) et les sorties du système par des fonctions « conséquences ». Pour un problème à K classes, une règle R_i est construite pour chaque modèle floue P_i :

$$\textbf{SI } \vec{x} \; est \; P_i \textbf{ ALORS } y_i^1 = a_i^1 \textbf{ et ... et } y_i^k = a_i^k \qquad [1]$$

Où $\vec{x}$ est un vecteur de caractéristiques de n dimensions, a_i^c est un poids qui exprime la participation du modèle P_i dans la description de la classe C. Chaque modèle flou P_i est défini par sa fonction d'appartenance.

Pour les systèmes d'apprentissage non-incrémental, où le système est construit en utilisant une base d'apprentissage avec un nombre suffisant d'exemples, des méthodes de classification supervisée où non-supervisée sont utilisées pour créer les modèles (les centres et les matrices de covariance) pour chaque classe.

Dans (Almaksour *et al.*, 2009), nous avons proposé une stratégie de clustering incrémental pour un SIF basée sur la détection des zones d'ambiguïté. Nous avons utilisé le rejet d'ambiguïté comme élément déclenchant pour la création d'un nouveau prototype. Nous avions pour objectif de limiter le nombre de prototypes dans le système en évitant de créer des prototypes dans des endroits où il n'y a pas de risque de confusion. Ceci élimine l'ajout des prototypes « inutiles » pour une classe spécifique où cette classe était déjà dominante. Bien que cette méthode ait obtenu de bonnes performances, il était difficile de trouver un seuil de rejet universel et optimal. L'inconvénient principal de cette méthode reste encore le trop grand nombre de prototypes pour certains problèmes, ce qui conduit à des systèmes « lourds » parce que trop complexes.

Une autre approche de clustering incrémental a été présentée dans (Angelov *et al.*, 2004), appelée *eClustering*. L'idée principale de l'approche proposée basée sur la définition du *potentiel* d'un point, qui représente la densité dans l'espace des données associée à ce point. Un exemple à fort potentiel est considéré comme un candidat pour être le centre d'un cluster. Le potentiel d'un exemple, qui a été présenté initialement dans (Yager *et al.*, 1993), peut être défini comme étant l'inverse de la somme

des distances entre cet exemple et tous les autres exemples. Une formule récursive (non-itératif, en-ligne) pour le calcul du potentiel de nouvelles données a été introduite dans (Angelov *et al.*, 2004). L'avantage principal de cette méthode de clustering incrémental est que ses paramètres (peu nombreux) sont indépendants du problème, et qu'il génère généralement beaucoup moins de prototypes que la stratégie de clustering guidée par l'ambiguïté.

Bien que la méthode eClustring peut générer un ensemble compact de règles lorsqu'elle est utilisée pour apprendre incrémentalement un SIF, la partie intrinsèque du système est moins efficace comparée à celle générée par la stratégie guidée par l'ambiguïté. Pour faire face à ce problème, les fonctions de conséquences du SIF doivent être améliorées en augmentant leur capacité discriminante. Pour cela, nous proposons d'utiliser un SIF de type Takagi-Sugeno d'ordre 1 en remplaçant la valeur constante dans la partie conséquence de la règle floue [1] par des fonctions linéaires. Plus de détails sur l'architecture du système sont présentés dans la section suivante.

3. Architecture du système

Comme mentionné précédemment, notre système est basé sur un système d'inférence floue de type Takagi-Sugeno d'ordre 1. Il se compose d'un ensemble de règles floues de la forme suivante :

$$\textbf{Règle}_i : \quad \textbf{SI } \vec{x} \; est \; P_i \; \textbf{ALORS } y_i^1 = l_i^1(\vec{x}) \; \textbf{et ... et } y_i^k = l_i^k(\vec{x}) \qquad [2]$$

où $l_i^m(\vec{x})$ représente les fonctions conséquences linéaires de la règle i pour la classe m :

$$l_i^m(\vec{x}) = \vec{\pi}_i^m \vec{x} = a_{i0}^m + a_{i1}^m x_1 + a_{i2}^m x_2 + ... + a_{in}^m x_n \qquad [3]$$

Pour trouver la classe de $\vec{x}$, son degré d'appartenance à chaque prototype floue $\beta_i(\vec{x})$ est calculé. Celui-ci est représenté dans notre système par une fonction à base radiale hyper-ellipsoïdale, caractérisée par un centre $\vec{\mu_i}$ et une matrice de covariance Q_i. Le degré d'activation est calculé en fonction de la distance de Mahalanobis du centre $d_{Q_i}(\vec{x}, \vec{\mu_i})$:

$$\beta_i(\vec{x}) = 1/(1 + d_{Q_i}(\vec{x}, \vec{\mu_i})) \qquad [4]$$

L'inférence floue de type somme-produit est ensuite utilisée pour calculer la sortie du système pour chaque classe :

$$y^m(\vec{x}) = \sum_{i=1}^{R} \beta_i(\vec{x}) \; l_i^m(\vec{x}) \qquad [5]$$

L'étiquette de la classe gagnante est donnée en trouvant la sortie maximale et en prenant l'étiquette de classe correspondante comme réponse :

$$classe(\vec{x}) = y = argmax \; y^m(\vec{x}) \qquad m = 1, .., k \qquad [6]$$

4. Modèle d'apprentissage incrémental proposé

4.1. *Clustering incrémental*

Dans un problème d'apprentissage incrémental en-ligne, les données d'apprentissage ne sont disponibles qu'au fur et à mesure. Le système doit donc être appris en utilisant les premières données arrivées, puis continuer à évoluer de manière transparente du point de vue utilisateur. Les algorithmes de clustering classiques nécessitent de connaître tous les données afin d'effectuer le clustering. Comme mentionné dans l'introduction, un critère très important d'une solution efficace d'apprentissage incrémental consiste à éviter (ou minimiser) l'accès aux données d'apprentissage précédentes. Nous avons donc besoin d'un algorithme capable de mettre à jour les clusters chaque fois que de nouvelles données sont disponibles, sans avoir besoin de tout reconstruire ni d'avoir accès aux données précédentes.

Quand on introduit un nouvel exemple d'apprentissage dans un mode d'apprentissage en-ligne, il pourra soit de renforcer l'information contenue dans les données précédentes et représentée par les clusters existants, soit apporter une nouvelle information suffisamment importante pour former un nouveau cluster ou modifier un cluster existant. L'importance d'un exemple dans le processus de clustering est évaluée par sa valeur de *potentiel*. Le potentiel d'un exemple est défini comme étant l'inverse de la somme des distances entre un exemple et tous les autres exemples de données (Yager *et al.,* 1993) :

$$P_k(x(k)) = \frac{1}{1 + \sum_{i=1}^{k-1} \|x(k) - x(i)\|^2} \quad [7]$$

Une méthode récursive pour le calcul du potentiel d'un nouvel exemple a été introduit dans (Angelov *et al.,* 2004). Cela constitue une solution prometteuse pour tout problème de clustering incrémental. La formule récursive évite la mémorisation de l'ensemble des données précédentes, mais conserve - en utilisant quelques variables - la distribution de densité dans l'espace. Le potentiel est défini de la façon suivante :

$$P_k(x(k)) = \frac{k-1}{(k-1)\alpha(k) + \gamma(k) - 2\zeta(k) + k - 1} \quad [8]$$

où

$$\alpha(k) = \sum_{j=1}^{n} x_j^2(k) \quad [9]$$

$$\gamma(k) = \gamma(k-1) + \alpha(k-1), \quad \gamma(1) = 0 \quad [10]$$

$$\zeta(k) = \sum_{j=1}^{n} x_j(k)\eta_j(k), \quad \eta_j(k) = \eta_j(k-1) + x_j(k-1), \quad \eta_j(1) = 0 \quad [11]$$

L'introduction d'un nouvel exemple affecte les valeurs du potentiel des centres des clusters existants, qui peuvent être mis à jour récursivement par l'équation suivante :

$$P_k(\mu_i) = \frac{(k-1)P_{k-1}(\mu_i)}{k-2+P_{k-1}(\mu_i)+P_{k-1}(\mu_i)\sum_{j=1}^{n}\|\mu_i - x(k-1)\|_j^2} \qquad [12]$$

Si le potentiel du nouvel exemple est plus élevé que le potentiel des centres existants, alors cet exemple sera le centre d'un nouveau cluster (et une nouvelle règle floue sera ajoutée dans le cas de notre SIF). Si l'exemple à fort potentiel est proche d'un centre existant μ_i, alors il le remplace et aucun nouveau cluster ne sera créé.

4.2. *Apprentissage incrémental des paramètres des fonctions conséquences linéaires*

Le problème de l'apprentissage des conséquences dans un SIF d'ordre 1 peut être résolu par la méthode des Moindres Carrés Récursifs Pondérés (MCRP) (Angelov *et al.*, 2008). Soit Π la matrice de tous les paramètres des conséquences linéaires du système :

$$\Pi = \begin{bmatrix} \vec{\pi}_1^1 & \vec{\pi}_1^2 & \dots & \vec{\pi}_1^m \\ \vec{\pi}_2^1 & \vec{\pi}_2^2 & \dots & \vec{\pi}_2^m \\ \dots & \dots & \dots & \dots \\ \vec{\pi}_r^1 & \vec{\pi}_r^2 & \dots & \vec{\pi}_r^m \end{bmatrix} \qquad [13]$$

où m représente le nombre de classes, et r est le nombre de règles floues ; Elle peut être récursivement estimée par :

$$\Pi_k = \Pi_{k-1} + C_k\psi_k(Y_k - \psi_k^T\Pi_{k-1}) \; ; \quad \Pi_1 = 0 \qquad [14]$$

$$C_k = C_{k-1} - \frac{C_{k-1}\psi_k\psi_k^T C_{k-1}}{1+\psi_k^T C_{k-1}\psi_k} \; ; \quad C_1 = \Omega I \qquad [15]$$

où $\psi_k = [\beta_1(\vec{x}_k)\vec{x}_k, \beta_2(\vec{x}_k)\vec{x}_k, ..., \beta_r(\vec{x}_k)\vec{x}_k]$ est le vecteur d'entrées pondéré par les niveaux d'activation des prototypes, I est la matrice identité et Ω est une constante (nombre positif grand), typiquement entre 100 et 10 000. Petites valeurs de Ω ralentissent l'apprentissage, mais un trop grand Ω peut empêcher les paramètres de converger correctement. La valeur doit donc être sélectionnée pour être un compromis entre les deux points. Ω = 1000 est adéquate pour la plupart des cas.

Quand une nouvelle règle est ajoutée au système, ses paramètres sont initialisés par la moyenne pondérée des paramètres des autres règles :

$$\Pi_k = \begin{bmatrix} \vec{\pi}_{1(k-1)}^1 & \vec{\pi}_{1(k-1)}^2 & \dots & \vec{\pi}_{1(k-1)}^m \\ \vec{\pi}_{2(k-1)}^1 & \vec{\pi}_{2(k-1)}^2 & \dots & \vec{\pi}_{2(k-1)}^m \\ \dots & \dots & \dots & \dots \\ \vec{\pi}_{r(k-1)}^1 & \vec{\pi}_{r(k-1)}^2 & \dots & \vec{\pi}_{r(k-1)}^m \\ \vec{\pi}_{(r+1)k}^1 & \vec{\pi}_{(r+1)k}^2 & \dots & \vec{\pi}_{(r+1)k}^m \end{bmatrix} \qquad [16]$$

où

$$\vec{\pi}^c_{(r+1)k} = \sum_{i=1}^{r} \beta_i(\vec{x}_k)\vec{\pi}^c_{i(k-1)} \quad [17]$$

En outre, la matrice de covariance C est étendue comme suit :

$$C_k = \begin{bmatrix} \rho \begin{bmatrix} C_{k-1} \end{bmatrix} & \begin{bmatrix} 0 \end{bmatrix} \\ \begin{bmatrix} 0 \end{bmatrix} & \begin{bmatrix} \Omega & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & \Omega \end{bmatrix} \end{bmatrix} \quad [18]$$

où $\rho = (r^2 + 1)/r^2$.

4.3. *Adaptation des prémisses*

Comme on peut le noter dans la section 4.1, la condition permettant d'avoir un fort potentiel est très dure, et elle est inversement proportionnelle au nombre croissant de données. Ainsi, on peut imaginer que le centre d'un cluster $\vec{\mu_i}$ qui n'est pas vraiment en position optimale étant donné l'historique des données présentées, ne sera pas modifié car il conserve la plus grande valeur de potentiel relativement aux autres données. Par conséquent, le processus de clustering incrémental de la partie prémisse d'un SIF ne sera pas en mesure de profiter des données qui n'ont pas un potentiel suffisamment élevé pour déplacer (ou déformer) les clusters existants.

Nous améliorons le processus de clustering incrémental (décrites dans la section 4.1) par un algorithme d'adaptation qui permet de modifier tous les modèles du SIF en les recentrant et les déformant pour chaque nouvelle donnée. Notre algorithme est inspiré par la méthode FLVQ (Fuzzy Learning Vector Quantization) (Chung *et al.*, 1994). Dans cette méthode, plus l'activation β_i de la prémisse de la règle i est loin de sa valeur objective β_i^*, plus le prototype doit être déplacé :

$$\Delta\vec{\mu}_i = \lambda * (\beta_i^* - \frac{\beta_i(\vec{x})}{\sum_{q=1}^{r} \beta_q(\vec{x})}) * (\vec{x} - \vec{\mu_i}) \quad [19]$$

où le paramètre d'adaptation λ est compris entre 0 et 1. Le score objectif β_i^* vaut 1 si l'exemple $\vec{x}$ et le prototype P_r sont de la même classe et 0 sinon. L'inconvénient principal d'utiliser directement cette méthode dans notre SIF est qu'il ne prend en considération que le score objectif de ce prototype et pas sa participation dans le score final de sortie. Par conséquent, nous avons étendu cet algorithme pour qu'il modifie tous les prototypes du système pour chaque nouvel exemple en fonction de leurs participations effectives au processus de reconnaissance (et non en fonction de leurs activations objectives). Ainsi, la mise à jour d'un prototype doit améliorer le score de

chaque classe, donc, le déplacement $\Delta\vec{\mu}_i$ doit être importante si le score de classe y^c est différent de sa valeur objective $\hat{y}^c$; plus la participation du prototype au score final de la classe y_i^c est élevé et plus la prémisse de la règle β_i sera fortement activée :

$$\begin{aligned}\Delta\vec{\mu}_i &= \lambda * \left(\beta_i(\vec{x}) \sum_{c=1}^{m} \left((\hat{y}^c(\vec{x}) - y^c(\vec{x}))\ y_i^c(\vec{x}) \right) \right) * (\vec{x} - \vec{\mu_i}) \\ &= \lambda * \left(\beta_i^2(\vec{x}) \sum_{c=1}^{m} \left((\hat{y}^c(\vec{x}) - y^c(\vec{x}))\ \vec{l}_i^c(\vec{x}) \right) \right) * (\vec{x} - \vec{\mu_i})\end{aligned} \quad [20]$$

où $\hat{y}^c$ vaut 1 si c est la classe de $\vec{x}$ et 0 sinon.

Comme mentionné dans la section 4.1, lorsque le potentiel du nouvel exemple est plus élevé que les potentiels des centres existants alors cet exemple sera le centre d'un nouveau prototype P_{r+1}, et la matrice de covariance Q_{R+1}, qui définit la zone d'influence (selon la distance de Mahalanobis), sera initialisée par une matrice proportionnelle à la matrice identité (ce qui se traduit par une forme hyper-sphérique).

Afin de mieux représenter la répartition du nuage de données qui ont incrémentalement formé le prototype, la forme du prototype qui est donnée par sa matrice de covariance doit être adapté et modifié en fonction de chaque nouvel exemple. Une formule récursive pour mettre à jour l'inverse de la matrice de covariance dans un contexte non-supervisé est donnée dans (De Backer *et al.*, 2001). Elle utilise les activations des prototypes dans le calcul de la matrice de covariance. Dans (Mouchere *et al.*, 2007), une nouvelle version de cette formule est présentée en utilisant, en plus des activations des prototypes, la participation du prototype à la reconnaissance de chaque classe, et l'erreur commise sur chaque classe. Sur la base de la formule présentée dans (Mouchere *et al.*, 2007), et en l'adaptant à notre architecture (SIF d'ordre 1), nous obtenons la formule récursive suivante pour mettre à jour l'inverse de la matrice de covariance Q_i :

$$Q_i^{-1} \Leftarrow \frac{Q_i^{-1}}{1-\alpha\delta_i} - \frac{\alpha\delta_i}{1-\alpha\delta_i} \cdot \frac{(Q_i^{-1}\vec{d}) \cdot (Q_i^{-1}\vec{d})^T}{1+\alpha\delta_i(\vec{d}^T Q_i^{-1}\vec{d})} \quad [21]$$

$$\delta_i = \beta_i^2(\vec{x}) \sum_{c=1}^{m} \left((\hat{y}^c(\vec{x}) - y^c(\vec{x}))\ \vec{l}_i^c(\vec{x}) \right) \quad [22]$$

avec $\vec{d} = \vec{x} - \vec{\mu_r}$ et α un facteur d'apprentissage.

4.4. *Stabilité de l'apprentissage des paramètres des fonctions conséquences*

L'une des propriétés favorables de la méthode des moindres carrés récursifs pondérés, est que elle converge vers la solution optimale pour chaque étape d'apprentissage. Mais, cette propriété n'est vraie que lorsque les poids (les activations des prémisses

dans notre cas) qui ont été associés aux exemples précédents restent inchangés. Dans notre système, les prototypes flous qui forment la partie prémisse sont dynamiques : ils peuvent être recentrés, déplacés ou déformés par la méthode de clustering incrémental (section 4.1) et l'algorithme d'adaptation (section 4.3). Par conséquent, les activations plus anciennes qui ont participé à l'apprentissage des paramètres des conséquences pour les données précédentes ne sont plus les mêmes. Cela rend l'estimation antérieure de ces paramètres incohérente avec le modèle actuel de prémisse.

Dans (Lughofer, 2008), l'auteur aborde ce problème de stabilité. Il propose l'idée d'introduire, après chaque modification de prémisse, un vecteur de correction pour les paramètres linéaires et une matrice de correction pour leur matrice de covariance, afin d'aller vers la solution optimale en fonction du degré de changement dans la partie prémisse. A notre connaissance, cette idée n'a jamais été formalisée étant donnée la complexité d'estimation de ces vecteurs et matrices de corrections. Ce problème complexe reste donc aujourd'hui toujours ouvert.

Pour faire face à ce problème de stabilité, nous proposons d'appliquer le clustering incrémental et l'algorithme d'adaptation sur la partie prémisse en mode « batch », au lieu de le refaire pour chaque exemple. De cette façon, nous continuons à appliquer l'algorithme de la mise à jour de paramètres de conséquences pour chaque exemple avec une structure fixe de prémisse. En outre, nous gardons les exemples de données dans une mémoire tampon de taille F. Ainsi, après l'introduction de F exemples de données, le clustering incrémental et l'algorithme d'adaptation sont appliqués pour chaque exemple de la mémoire tampon. Puis, un réajustement des paramètres des conséquences est fait en appliquant la méthode MCRP sur les exemples de la mémoire (avec la structure modifiée de prémisse). Après ce processus de réajustement, la mémoire tampon est vidée et la structure des prémisses est gelée, alors que le processus de la mise à jour des paramètres des conséquences se poursuit pour chaque nouvel exemple.

L'algorithme complet de notre modèle d'apprentissage incrémental est résumé dans Algorithme 1.

5. Evaluation

Dans cette section, nous évaluons la performance de notre système d'apprentissage dans deux contextes différents. Dans le premier, nous utilisons un problème de classification relativement facile, de sorte que plusieurs systèmes avec des architectures différentes peuvent tendre vers un même taux de reconnaissance élevé, mais le but de cette expérience est de voir à quelle vitesse ces taux s'améliorent au début du processus d'apprentissage incrémental (où seulement peu de données d'apprentissage sont disponibles). D'autre part, nous visons dans la seconde expérience à tester notre système sur un problème de benchmark plus difficile, afin de montrer l'évolution de ses performances à long terme et de comparer les taux obtenus avec ceux de quelques méthodes d'apprentissage non-incrémental connues.

Algorithme 1 : Algorithme d'apprentissage incrémental de SIF d'ordre 1

pour chaque *nouvel exemple* $\vec{x}$ **faire**
  si $\vec{x}$ *est le premier exemple d'une nouvelle classe* **alors**
    créer un nouveau prototype autour de $\vec{x}$;
    initialiser son potentiel par 1;
    ajouter une nouvelle règle floue au système;
    étendre la matrice des paramètres des consequences comme dans [16] et [17];
    mettre à jour et étendre la matrice de covariance comme dans [18];
  sinon
    calculer les activations des règles floues par [4];
    déterminer la classe de $\vec{x}$ par [6];
    obtenir la classe effective de $\vec{x}$;
    calculer le potentiel de $\vec{x}$ par [8];
    mettre à jour les potentiels des centres des prototypes existants en utilisant [12];
    si $P(\vec{x}) > P_k(\vec{\mu}_i) \ \ \forall i \in [1, R]$ **alors**
      si $\vec{x}$ *est près d'un centre existant* $\vec{\mu}_i$ **alors**
        mettre $\vec{x}$ le nouveau centre du prototype P_i et garder les mêmes consequences que la *règle*$_i$;
      sinon
        créer un nouveau prototype autour de $\vec{x}$;
        initialiser son potentiel par 1;
        ajouter une nouvelle règle floue au système;
        étendre la matrice des paramètres des consequences comme dans [16] et [17];
        mettre à jour et étendre la matrice de covariance comme dans [18];
      fin
    fin
    mettre à jour les paramètres des consequences par [14] et [15];
    ajouter $\vec{x}$ à la mémoire tampon;
    si *mémoire tampon est pleine* **alors**
      pour chaque *exemple* $\vec{x}_b$ *dans la mémoire tampon* **faire**
        Appliquer l'adaptation des prémisses en fonction de $\vec{x}_b$ par [20] et [21];
      fin
      pour chaque *exemple* $\vec{x}_b$ *dans la mémoire tampon* **faire**
        mettre à jour les paramètres des consequences en fonction de $\vec{x}_b$ par [14];
      fin
      vider la mémoire tampon;
    fin
  fin
fin

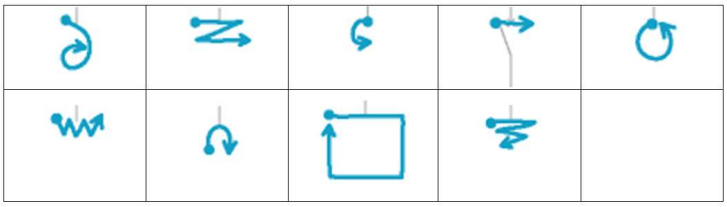

Figure 1. *Les symboles électriques manuscrits utilisés*

5.1. *Première expérience*

Nous avons mené cette expérience sur un ensemble de données de neuf symboles électriques manuscrits en-ligne (figure 1). Chaque symbole est décrit par un ensemble de 21 caractéristiques. Nous exploitons 12 tests différents sur 12 bases mono-scripteurs saisies par des personnes différentes sur un Tablet PC. Les résultats présentés sont la moyenne de 12 essais. Chaque scripteur a saisi 50 exemples de chaque symbole, soit 450 symboles dans chaque base mono-scripteur. Nous avons utilisé trois quarts de chaque base pour le processus d'apprentissage incrémental, et le dernier quart pour estimer l'évolution de la performance du système pendant le processus d'apprentissage. Un nouvel exemple de chaque classe est présenté au système entre chaque évaluations consécutives.

Quatre modèles flous d'apprentissage incrémental (avec des architectures différentes ou différents algorithmes d'apprentissage) sont comparés dans cette expérience :

– Modèle I : présenté dans (Almaksour *et al.*, 2009), et brièvement décrit dans la section 2.

– Modèle II : SIF d'ordre 1 avec un clustering incrémental à base de potentiel et la méthode MCRP.

– Modèle III : modèle II + adaptation de prémisse pour chaque exemple.

– Modèle IV : modèle II + adaptation de prémisse en mode « batch » + réajustement des paramètres des conséquences.

Les paramètres λ, α, Ω et F sont fixés à 0.005, 0.001, 1000 et 30 respectivement. Afin d'avoir des valeurs référentielles dans l'évaluation du taux de reconnaissance sur l'ensemble de données utilisées, nous avons appris deux classifieurs classiques (non-incrémentaux) en utilisant la totalité de la base d'apprentissage, et nous avons mesuré leurs performances sur la base de test. Nous avons choisi un classifieur de type Multi-Layer Perceptron (MLP) et un autre de type K-plus proches voisins (K-NN, avec K = 5). Nous remarquons sur la figure 2 qu'en utilisant le modèle IV, la performance s'améliore rapidement avec très peu de données. Par exemple, un taux de reconnaissance de plus de 91 % est atteint après seulement 5 exemples par classe, et il monte rapidement et dépasse 94 % après 10 exemples. En comparant la performance du modèle

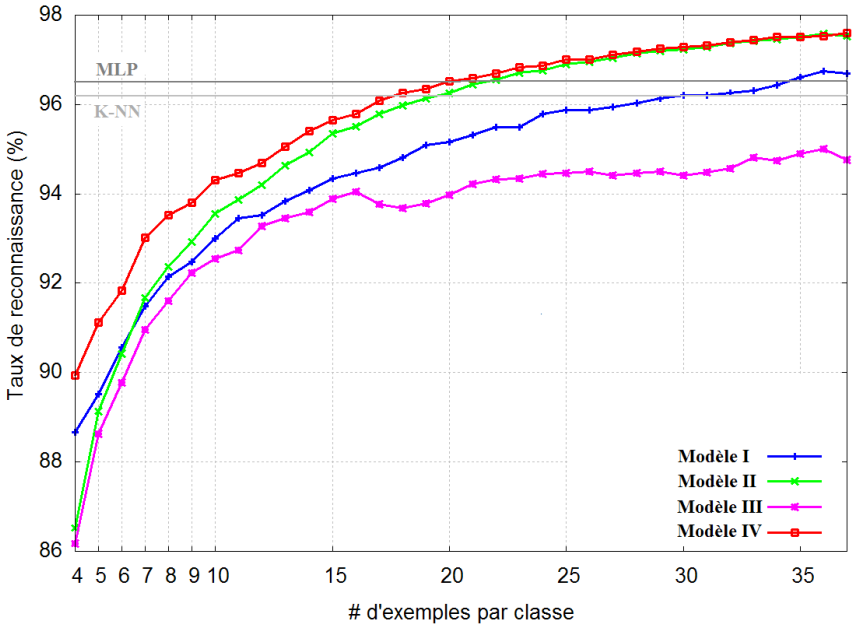

Figure 2. *Evolution du taux de reconnaissance au début du processus d'apprentissage incrémental.*

IV et celle du modèle II dans la figure 2, nous voyons que l'intégration de l'adaptation des prémisses dans le modèle contribue à booster et à accélérer le processus d'apprentissage, surtout au début quand peu de données d'apprentissage ont été introduites. Cette propriété est très importante dans notre contexte applicatif, car l'utilisateur doit être en mesure d'utiliser le nouveau geste dès que possible. Il est donc important que la performance du système devienne satisfaisante aussi rapidement que possible. Il n'est pas étonnant que les courbes des deux modèles se rencontrent après l'introduction d'un nombre significatif d'exemples, c'est parce que le problème de classification proposée est relativement facile, et le modèle II, même en l'absence d'une structure de prémisse optimale, peut atteindre la performance du modèle IV. Nous remarquons également sur la même figure l'instabilité de la performance du modèle III, qui est due à la perturbation continue des paramètres des conséquences à cause de l'adaptation de prémisse pour chaque nouvel exemple. Le modèle I a une bonne performance au début de l'apprentissage, mais comme nous l'avons mentionné dans la section 2, c'est au prix de la création de beaucoup de prototypes dans le système. Par exemple, il y a 90 prototypes flous dans le modèle I après l'introduction de 315 exemples (9x35), tandis que pour les trois autres modèles il y a en moyenne moins de 11 prototypes. Un nombre élevé de prototype peut accélérer la performance au début, mais le système devient rapidement trop « lourd » et donc moins à même d'apprendre et de s'adapter.

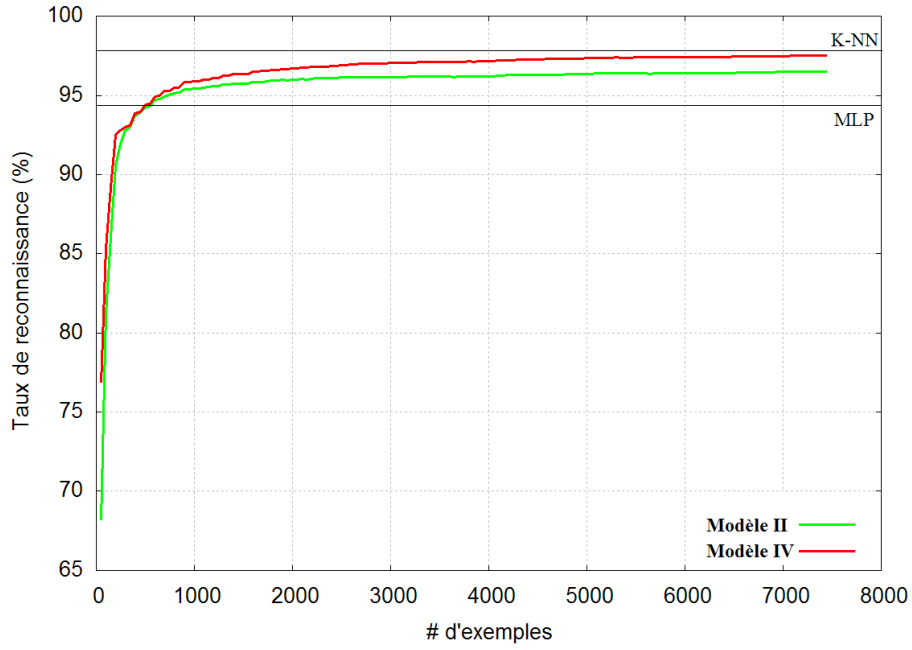

Figure 3. *Evolution du taux de reconnaissance dans un processus d'apprentissage incrémental à long terme.*

5.2. *Seconde expérience*

Nous avons utilisé pour cette expérience, la base « Pen-based recognition » de UCI-repository. La base contient 7494 exemples d'apprentissage et 3498 exemples de test. Chaque exemple est représenté par 16 caractéristiques. Elle contient 10 classes pour les 10 chiffres. Nous pouvons globalement remarquer de la figure 3 que la performance du modèle d'apprentissage incrémental proposé peut atteindre ou dépasser la performance des modèles d'apprentissage classique connus. Nous remarquons également la stabilité du processus d'apprentissage de façon qu'il n'y ait presque aucune chute de la performance du système tout au long du processus d'apprentissage.

6. Conclusion

Dans le contexte des systèmes de reconnaissance de gestes manuscrits, nous avons présenté dans ce papier un nouvel algorithme d'apprentissage incrémental basé sur un système d'inférence floue d'ordre 1. Grace à cet algorithme, le système de reconnaissance peut commencer à apprendre à partir de zéro et avec peu de données d'apprentissage. Il peut aussi s'améliorer et s'adapter aux données nouvellement dis-

ponibles. Les résultats ont montré qu'un taux de reconnaissance raisonnable (environ 91 %) peut être obtenu dès l'introduction de 5 exemples par classe. Pour les travaux futurs, un accent particulier sera mis sur la recherche d'une estimation d'un vecteur de correction des conséquences linéaires après chaque adaptation de prémisse, ce qui peut améliorer la performance du système et éliminer le besoin d'une mémoire tampon. Une autre perspective liée au contexte de l'application est de concevoir et mettre en place des techniques de synthèse d'exemples artificielles à partir de gestes manuscrits originaux, afin d'accélérer encore le processus d'apprentissage lorsque très peu de données sont disponibles.

7. Bibliographie

Aha D. W., Kibler D., Albert M. K., « Instance-Based Learning Algorithms », *Mach. Learn.*, vol. 6, n° 1, p. 37-66, 1991.

Almaksour A., Anquetil E., « Fast Incremental Learning Strategy Driven by Confusion Reject for Online Handwriting Recognition », *Tenth International Conference on Document Analysis and Recognition (ICDAR 2009)*, p. 81-85, 2009.

Almaksour A., Mouchère H., Anquetil E., « Fast Online Incremental Learning with Few Examples For Online Handwritten Character Recognition », *Proceedings of the Eleventh International Conference on Frontiers in Handwriting Recognition (ICFHR'08)*, Montréal, Québec, Canada, p. 623-628, August, 2008.

Angelov P., Filev D., « An approach to online identification of Takagi-Sugeno fuzzy models », *Systems, Man, and Cybernetics, Part B : Cybernetics, IEEE Transactions on*, vol. 34, n° 1, p. 484-498, Feb., 2004.

Angelov P., Lughofer E., Zhou X., « Evolving fuzzy classifiers using different model architectures », *Fuzzy Sets Syst.*, vol. 159, n° 23, p. 3160-3182, 2008.

Carpenter G. A., Grossberg S., « The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network », *Computer*, vol. 21, n° 3, p. 77-88, 1988.

Carpenter G., Grossberg S., Markuzon N., Reynolds J., Rosen D., « Fuzzy ARTMAP : A neural network architecture for incremental supervised learning of analog multidimensional maps », *IEEE Transactions on Neural Networks*, 1992.

Chung F., Lee T., « Fuzzy Competitive Learning », *IEEE Transaction on Neural Network*, vol. 7, n° 3, p. 539-551, 1994.

De Backer S., Scheunders P., « Texture Segmentation by Frequency-Sensitive Elliptical Competitive Learning », *Image and Vision Computing*, vol. 19, n° 9–10, p. 639-648, 2001.

Gary G. Yen P. M., « An effective neuro-fuzzy paradigm for machinery condition health monitoring », *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 31-4, p. 523 - 536, 2001.

Jang J.-S., « ANFIS : adaptive-network-based fuzzy inference system », *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 23, p. 665-685, 1993.

Jr. J. J. L., Zeleznik R. C., « A Practical Approach for Writer-Dependent Symbol Recognition Using a Writer-Independent Symbol Recognizer », *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, n° 11, p. 1917-1926, 2007.

Littlestone N., « Redundant noisy attributes, attribute errors, and linear-threshold learning using winnow », *COLT '91 : Proceedings of the fourth annual workshop on Computational learning theory*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 147-156, 1991.

Littlestone N., Warmuth M. K., « The weighted majority algorithm », *Inf. Comput.*, vol. 108, n° 2, p. 212-261, 1994.

Lughofer E., « FLEXFIS : A Robust Incremental Learning Approach for Evolving TakagiSugeno Fuzzy Models », *Fuzzy Systems, IEEE Transactions on*, vol. 16, n° 6, p. 1393-1410, Dec., 2008.

Minku F. L., Inoue H., Yao X., « Negative correlation in incremental learning », *Natural Computing : an international journal*, vol. 8, n° 2, p. 289-320, 2009.

Mouchere H., Anquetil E., Ragot N., « On-line writer adaptation for handwriting recognition using fuzzy inference systems », *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, vol. 21(1), p. 99-116, 2007.

Polikar R., Udpa L., Udpa S., Honavar V., « Learn++ : An Incremental Learning Algorithm for Supervised Neural Networks », *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 31, p. 497-508, 2001.

Reinke R. M. R., « Incremental learning of concept descriptions : A method and experimental results », *Hayes, J., Michie, D., Richards, J., eds. : Machine Intelligence*, vol. 11, p. 263288, 1988.

Sadri J., Suen C. Y., Bui T. D., « A New Clustering Method for Improving Plasticity and Stability in Handwritten Character Recognition Systems », *Pattern Recognition, International Conference on*, vol. 2, p. 1130-1133, 2006.

Takagi T., Sugeno M., « Fuzzy identification of systems and its applications to modeling and control », *IEEE TSMC*, vol. 15, n° 1, p. 116-132, 1985.

Yager R. R., Fileu D. P., « Learning of fuzzy rules by mountain clustering », vol. 2061, SPIE, p. 246-254, 1993.