Étude des techniques d'oubli dans les moindres carrés récursifs pour l'apprentissage incrémental de systèmes d'inférence floue évolutifs : application à la reconnaissance de formes

Manuel Bouillon*, Eric Anquetil*
Abdullah Almaksour*

* INSA de Rennes, Avenue des Buttes de Coësmes, F-35043 Rennes
Université Européenne de Bretagne, France
CNRS, UMR IRISA, Campus de Beaulieu, F-35042 Rennes
{manuel.bouillon, eric.anquetil, abdullah.almaksour}@irisa.fr,
http://www.irisa.fr/intuidoc/

Résumé. Cet article étudie les possibilités d'utilisation d'oubli dans l'apprentissage incrémental en-ligne de classifieurs évolutifs basés sur des systèmes d'inférence floue. Pour cela, nous étudions différentes possibilités, existant dans la littérature dédiée au contrôle, pour introduire de l'oubli dans l'algorithme des moindres carrés récursifs. Nous présentons l'impact de ces différentes techniques dans le contexte de l'apprentissage incrémental de classifieurs évolutifs en environnement non stationnaire. Ces approches sont évaluées, pour l'optimisation des systèmes d'inférence floue, sur la problématique de la reconnaissance de gestes manuscrits sur surface tactile.

1 Introduction

L'objectif de ces travaux est d'obtenir un classifieur évolutif pour faciliter les interactions homme machine sur interface tactile (tablette, smartphone, tableau interactif, etc.). Autrement dit, nous souhaitons que le classifieur soit capable de s'adapter à la manière d'écrire et de dessiner de l'utilisateur. Cette manière de dessiner va probablement évoluer au fur et à mesure que l'utilisateur va s'habituer à utiliser l'interface tactile. L'utilisateur commence par dessiner lentement et attentivement ses gestes lorsqu'il est novice, alors qu'il les fait d'une manière plus fluide et plus rapide quand il devient expert. Il faut alors que le classifieur s'adapte et suive cette évolution. Il est également souhaitable que le classifieur supporte l'ajout de nouvelles classes « à la volée » pour permettre à l'utilisateur de personnaliser et d'adapter l'application à ses besoins. Pour cela, il est nécessaire de disposer d'un système performant et réactif aux changements de son environnement.

Plusieurs systèmes de classification évolutifs sont apparus ces dernières années pour répondre au besoin de classifieurs fonctionnant en environnement non stationnaire. Ils utilisent des algorithmes d'apprentissage incrémental pour améliorer leurs performances pendant leur

utilisation et s'adapter à tout changement. Parmi ces classifieurs évolutifs, les systèmes d'inférence floue (SIF) se distinguent par leurs excellentes performances (Angelov et Zhou, 2008), leur capacité à mettre en œuvre un apprentissage incrémental et leur bon comportement face à l'ajout de classe. Dans la plupart de ces approches, l'apprentissage incrémental des conclusions repose sur l'algorithme des moindres carrés récursifs. Or cet apprentissage incrémental lié aux moindres carrés récursifs à l'inconvénient de perdre ses capacités d'adaptation avec le temps. Cet article étudie les différentes possibilités existantes, dans la littérature dédiée au contrôle, pour introduire de l'oubli dans l'algorithme des moindres carrés récursifs, dans le contexte de la classification à base de SIF. L'objectif est de préserver dans le temps les capacités d'adaptation des SIF. Ces différentes techniques sont transposées dans le système d'inférence floue *Evolve* (Almaksour et Anquetil, 2011), puis sont comparées pour la reconnaissance de gestes manuscrits.

La section 2 présente l'architecture et l'apprentissage incrémental des systèmes d'inférence floue. Plusieurs approches pour intégrer de l'oubli dans l'algorithme des moindres carrés récursifs sont ensuite présentées section 3. Ces différentes techniques sont évaluées et comparées sur la problématique de reconnaissance de gestes manuscrits en section 4. Enfin, la section 5 conclue et discute des perspectives de ces travaux.

2 Systèmes d'inférence floue d'ordre un

Cette section présente l'architecture de notre système d'inférence floue (SIF) *Evolve*. Nous détaillons ensuite l'algorithme des moindres carrés récursifs utilisé pour son apprentissage. Enfin, nous expliquons les limites de cet algorithme ainsi que la nécessité d'intégrer de l'oubli.

2.1 Architecture du système d'inférence floue

Nous travaillons ici avec des systèmes d'inférence floue (SIF) d'ordre un – dit de Takagi-Sugeno – qui offrent de bonnes performances face à des problèmes de classifications complexes. De plus, ils supportent bien l'ajout de nouvelles classes « à la volée » et peuvent facilement être entrainés de manière incrémentale en temps réel. Des classifieurs évolutifs basés sur des SIF ont été proposés par Angelov et Zhou (2008) et par Almaksour et Anquetil (2011).

Un système d'inférence floue d'ordre un se compose d'un ensemble de règles floues de la forme suivante :

$$\textbf{Règle}^{(i)} : \textbf{SI } \mathbf{x} \text{ appartient à } C^{(i)} \textbf{ ALORS } \hat{\mathbf{y}}^{(i)} = (l_1^{(i)}(\mathbf{x}) ; \ldots ; l_c^{(i)}(\mathbf{x})) \tag{1}$$

où $\mathbf{x} \in \mathbb{R}^n$ le vecteur des caractéristiques et $\hat{\mathbf{y}}^{(i)} \in \mathbb{R}^c$ le vecteur d'appartenance au c différentes classes.

Les prémisses des règles sont définies par l'appartenance floue à des prototypes $C^{(i)}$ – des regroupements (*clusters*) dans l'espace d'entrée – qui sont caractérisés par un centre $\mu^{(i)}$ et une matrice de covariance $\Sigma^{(i)}$. Le degré d'appartenance floue $\beta^{(i)}(\mathbf{x})$ est calculé par une fonction à base radiale hyper-ellipsoïdale, en fonction de la distance de Mahalanobis $d_{\Sigma^{(i)}}(\mathbf{x}, \mu^{(i)})$ entre $\mathbf{x}$ et $C^{(i)}$:

$$\beta^{(i)}(\mathbf{x}) = \frac{1}{1 + d_{\Sigma^{(i)}}(\mathbf{x}, \mu^{(i)})} \tag{2}$$

Les conclusions des règles donnent l'appartenance floue aux différentes classes, et pour les SIF d'ordre un, ces degrés d'appartenance sont obtenus par des fonctions linéaires des entrées :

$$\hat{\mathbf{y}}^{(i)}(\mathbf{x}) = (l_1^{(i)}(\mathbf{x})\,;\,...\,;\,l_c^{(i)}(\mathbf{x})) \tag{3}$$

où $l_k^{(i)}(\mathbf{x})$ représente la fonction conséquence linéaire de la règle i pour la classe k :

$$l_k^{(i)}(\mathbf{x}) = \theta_k^{(i)\,\top}\mathbf{x} = \theta_{k,1}^{(i)}x_1 + ... + \theta_{k,n}^{(i)}x_n \tag{4}$$

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

$$\hat{\mathbf{y}}(\mathbf{x}) = \sum_{i=1}^{r} \beta^{(i)}(\mathbf{x}) \cdot \hat{\mathbf{y}}^{(i)}(\mathbf{x}) \tag{5}$$

où r représente le nombre de règles floues. La classe de $\mathbf{x}$ est choisie comme étant l'étiquette correspondant à la composante maximale de la sortie du système $\hat{\mathbf{y}}(\mathbf{x}) = (\hat{y}_1(\mathbf{x})\,;\,...\,;\,\hat{y}_c(\mathbf{x}))$:

$$\text{classe}(\mathbf{x}) = \arg\max\ \hat{y}_k(\mathbf{x}) \quad k = 1, ..., c \tag{6}$$

2.2 Apprentissage 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 de l'utilisateur, lorsque les données suivantes arrivent.

Ainsi, les prototypes sont adaptés incrémentalement à l'arrivée de chaque nouvelle donnée. Ce processus d'adaptation permet de mettre à jour les centres des prototypes en fonction de chaque nouvelle donnée d'apprentissage disponible, et de recalculer de manière récursive les matrices de covariances des prototypes. Un algorithme de *clustering* incrémental est utilisé pour créer de nouveaux prototypes, et donc de nouvelles règles, lorsque que cela est opportun.

Le problème de l'apprentissage incrémental des conséquences dans un SIF d'ordre un peut être résolu par la méthode des Moindres Carrés Récursifs (MCR) pondérés par les activations des règles (Angelov et Zhou, 2008). Soit $\Theta^{(i)}$ la matrice de tous les paramètres des conséquences linéaires de la règle i :

$$\Theta^{(i)} = \begin{bmatrix} \theta_{1,1}^{(i)} & \dots & \theta_{1,c}^{(i)} \\ \vdots & & \vdots \\ \theta_{n,1}^{(i)} & \dots & \theta_{n,c}^{(i)} \end{bmatrix} \tag{7}$$

où c représente le nombre de classes, et n la taille du vecteur des caractéristiques. Ces matrices peuvent être récursivement apprises :

$$\Theta_t^{(i)} = \Theta_{t-1}^{(i)} + P_t^{(i)}\mathbf{x}_t\beta^{(i)}(\mathbf{x}_t)(\mathbf{y}_t - \mathbf{x}_t^\top\Theta_{t-1}^{(i)})\,; \quad \Theta_0^{(i)} = 0 \tag{8}$$

Où les matrices de covariances $P^{(i)} = (\sum_{k=1}^{t}\mathbf{x}_k \cdot \mathbf{x}_k^\top)^{-1}$ sont également mises à jour récursivement :

$$P_t^{(i)} = P_{t-1}^{(i)} - \frac{P_{t-1}^{(i)}\mathbf{x}_t\mathbf{x}_t^\top P_{t-1}^{(i)}}{\frac{1}{\beta^{(i)}(\mathbf{x}_t)} + \mathbf{x}_t^\top P_{t-1}^{(i)}\mathbf{x}_t}\,; \quad P_0^{(i)} = 100 \cdot Id \tag{9}$$

2.3 Nécessité et principe de l'oubli

Plus le nombre de données d'apprentissage augmente, plus les modifications apportées aux conclusions par l'algorithme des MCR seront faibles. En effet, le poids accordé à chaque donnée étant équivalent, plus le nombre de données augmente, plus le poids de chaque donnée individuelle est faible. Cette propriété est illustrée par le fait que la matrice de covariance décroit avec le temps : $P_t < P_{t-1}$. Il est dit que le gain de l'algorithme tend vers zéro.

Cette propriété a pour conséquence de réduire la réactivité du système – la capacité d'apprentissage de nouveautés – avec le temps. Ainsi, au bout d'un certain temps, l'algorithme ne sera plus capable de s'adapter aux changements dans les données, ni d'apprendre de nouvelles classes (dans un temps raisonnable). Il est donc nécessaire de limiter le poids des anciennes données, autrement dit d'introduire de l'oubli dans l'algorithme des moindres carrés récursifs, pour maintenir les capacités d'apprentissage du système.

L'introduction d'oubli dans l'algorithme des moindres carrés récursifs est un sujet qui a été beaucoup étudié dans la littérature dédiée au contrôle de systèmes physiques complexes (Lughofer et Angelov, 2011). Pour cela, il faut travailler sur la matrice de covariance qui représente la distribution des données dans l'algorithme des moindres carrés récursifs. Le principe de l'oubli est d'introduire une mise à jour temporelle de cette matrice :

$$\bar{P}_t = \mathcal{F}(P_t) \tag{10}$$

en plus de la mise à jour d'observation (Equation 9). $\mathcal{F}(\cdot)$ est la fonction d'oubli et cette mise à jour temporelle représente alors l'augmentation de l'incertitude entre deux observations consécutives. Par conséquent, cette fonction doit être telle que : $\mathcal{F}(p) > p$.

Cette formulation de l'oubli – dite bayésienne – permet d'unifier les différentes techniques d'oubli pour l'algorithme des moindres carrés récursifs (Kulhavy et Zarrop, 1993). Chaque technique est alors caractérisée par sa fonction d'oubli $\mathcal{F}(\cdot)$.

La propriété la plus importante d'un algorithme d'estimation récursif avec oubli est que sa matrice de covariance reste bornée (Salgado et al., 1988), (Parkum et al., 1992). En effet, sans borne inférieure à la matrice de covariance, le gain de l'algorithme va tendre vers zéro et le système va perdre ses capacités d'adaptation. Sans borne supérieure, la matrice de covariance peut exploser sous l'effet de d'oubli, et engendrer des instabilités qui mènent à l'écroulement du système.

3 Différentes formes d'oubli

Cette section présente les principales techniques pour introduire de l'oubli dans l'algorithme des moindres carrés récursifs. La plus simple est l'utilisation d'un facteur d'oubli exponentiel, mais elle est sujette au problème d'explosion de la matrice de covariance (*covariance « wind-up » problem*). Plusieurs autres techniques ont été proposées pour tenter de pallier ce problème comme utiliser un facteur d'oubli variable ou de l'oubli directionnel.

3.1 Facteur d'oubli constant

Le principe du facteur d'oubli exponentiel est d'introduire une pondération exponentiellement décroissante des anciennes données dans le calcul de la matrice de covariance :

$$P_t = (\sum_{k=1}^{t} \lambda^{t-k} \cdot \mathbf{x}_k \cdot \mathbf{x}_k^\top)^{-1} = (\lambda^{t-1}\mathbf{x}_1^\top\mathbf{x}_1 + \cdots + \lambda\mathbf{x}_{t-1}^\top\mathbf{x}_{t-1} + \mathbf{x}_t^\top\mathbf{x}_t)^{-1} \tag{11}$$

soit de manière récursive :

$$P_t^{-1} = \lambda P_{t-1}^{-1} + \mathbf{x}_t^\top\mathbf{x}_t \tag{12}$$

où $\lambda \in [0;1]$ est le facteur d'oubli exponentiel (typiquement $\lambda \in [0.9;0.99]$). Par conséquent, le poids des anciennes données va être plafonné d'une part, et l'apprentissage des conclusions va se concentrer sur les dernières données d'autre part. C'est un moyen simple – en apparence – de maintenir les capacités d'apprentissage du système. Cette pondération exponentielle des données se traduit par la fonction d'oubli :

$$\mathcal{F}(p) = \frac{p}{\lambda} \tag{13}$$

Cependant, le choix de ce facteur d'oubli est complexe. Si le facteur d'oubli est trop faible, la matrice de covariance va tendre vers zéro et l'algorithme va perdre ses capacités d'adaptation. Si le facteur d'oubli est trop fort, la matrice de covariance va exploser et l'algorithme va devenir instable jusqu'à provoquer l'effondrement du système. Ce phénomène – appelé *covariance « wind-up »* – est dû au fait que l'on oubli davantage d'information que l'on en reçoit.

Pour éviter ce problème, il est nécessaire d'ajuster le facteur d'oubli à la quantité d'information nouvellement disponible. C'est le principe du facteur d'oubli variable dans le temps.

3.2 Facteur d'oubli variable

Le principe du facteur d'oubli variable dans le temps est d'ajuster la quantité d'information oubliée à la quantité d'information apprise. Pour cela, il faut disposer d'une mesure d'information ι_t que l'on va maintenir constante au cours du temps. Le facteur d'oubli est ensuite calculé $\lambda_t^{(i)} = \iota_t^{(i)}/\iota_{t-1}^{(i)}$ pour obtenir la fonction d'oubli :

$$\mathcal{F}_t^{(i)}(p) = \frac{p}{\lambda_t^{(i)}} = p \cdot \frac{\iota_{t-1}^{(i)}}{\iota_t^{(i)}} \tag{14}$$

Fortescue et al. (1981) proposent ainsi d'utiliser l'erreur quadratique *a posteriori* (pondérée) comme métrique :

$$\iota_t^{(i)} = \beta^{(i)}(\mathbf{x}_t)(\mathbf{y}_t - \mathbf{x}_t^\top\Theta_t^{(i)}) \tag{15}$$

Une autre mesure possible, reflétant la quantité d'information acquise par l'algorithme, est la trace de la matrice de covariance :

$$\iota_t^{(i)} = \text{trace}(P_t^{(i)}) \tag{16}$$

Un tel facteur d'oubli variable permet à l'algorithme de conserver un gain non nul au cours du temps et le système peut ainsi s'adapter à une évolution des données. Cette approche permet

aussi d'éviter l'explosion de la matrice de covariance, mais seulement si l'information est bien répartie selon les dimensions de l'espace d'entrée. Dans le cas contraire, certains éléments de la matrice de covariance peuvent tendre vers zéro pendant que d'autres explosent. Pour éviter cela, il faut non seulement faire varier le facteur d'oubli dans le temps mais également dans l'espace, c'est ce qu'on appelle l'oubli directionnel.

3.3 Oubli directionnel

L'oubli directionnel – *Directional Forgetting (DF)* – a été proposé par Hägglund (1985) et Kulhavy (1987) puis amélioré par Bittanti et al. (1990) et Cao et Schwartz (2000). C'est une technique qui fait varier l'oubli dans le temps mais aussi dans l'espace en introduisant de l'oubli seulement dans les directions où de l'information arrive.

La fonction d'oubli utilisée est donc paramétrée par la donnée courante :

$$\mathcal{F}_t^{(i)}(p) = p - (1 - \gamma_t^{(i)})\, \beta^{(i)}(\mathbf{x}_t)\mathbf{x}_t^\top \mathbf{x}_t \tag{17}$$

$$\gamma_t^{(i)} = \lambda_t^{(i)} - \frac{1 - \lambda_t^{(i)}}{\mathbf{x}_t^\top P_{t-1}^{(i)} \mathbf{x}_t} \tag{18}$$

L'avantage de cette technique est qu'elle maintient la matrice de covariance bornée, et garantit ainsi les performances du système. Dans les directions où beaucoup d'information arrive, on oublie beaucoup ; mais on oublie peu dans les directions qui sont peu excitées.

4 Résultats expérimentaux

Dans cette section, nous étudions l'impact de ces différentes approches d'oubli dans le cadre de l'apprentissage incrémental de systèmes d'inférence floue pour la classification.

4.1 Protocole d'évaluation

En partant du classifieur évolutif *Evolve* (Almaksour et Anquetil, 2011), nous avons implémenté les techniques d'oubli constant : *Evolve F* (*Forgetting*) ; d'oubli variable : *Evolve VF* (*Variable Forgetting*) ; et d'oubli directionnel : *Evolve DF* (*Directional Forgetting*) avec différents paramètres :

1. oubli constant avec $\lambda = 0.99$: *Evolve F (0.99)* ;
2. oubli constant avec $\lambda = 0.995$: *Evolve F (0.995)* ;
3. oubli constant avec $\lambda = 0.999$: *Evolve F (0.999)* ;
4. oubli variable avec $\iota_t^{(i)} = \text{trace}(P_t)$: *Evolve VF (trace)* ;
5. oubli variable avec $\iota_t^{(i)} = \beta_t^{(i)}(y_t - x_t^\top \Theta_t^{(i)})$: *Evolve VF (error)* ;
6. oubli directionnel avec $\lambda = 0.85$: *Evolve DF (0.85)*.

Nous utilisons la base de données ILGDB[1] (Renau-Ferrer et al., 2012) avec les caractéristiques HBF49 (Delaye et Anquetil, 2013) fournies. Cette base de donnée contient 6629 gestes

[1]Disponible gratuitement à l'adresse `http ://www.irisa.fr/intuidoc/ILGDB.html`

mono-stroke, appartenant à 21 classes, qui ont été saisis par 38 scripteurs dans un environnement immersif. Cette base de donnée est très intéressante pour plusieurs raisons. Tout d'abord, les gestes sont ordonnés chronologiquement suivant leur ordre de saisie ce qui permet de voir l'évolution de la manière de dessiner des scripteurs avec le temps. Ensuite, les fréquences des différentes classes varies, de 5 à 17 exemples par classe (par scripteur). Enfin, pour toute une partie de la base, les gestes des 21 classes ont été définis par les utilisateurs eux-mêmes. Ces caractéristiques font que cette base de données contient des gestes très réalistes et représentatifs de l'utilisation d'un classifieur en-ligne. En revanche, l'inconvénient de cette base est son faible nombre de gestes par scripteur (moins de 180) et par classe pour chaque scripteur (entre 5 et 17 gestes). Des exemples des gestes inventés par les scripteurs sont présentés Figure 1.

FIG. 1 – *Exemples de gestes – ILGDB groupe 1 (gestes libres).*

Pour que l'évaluation soit représentative de l'utilisation d'un système évolutif, nous utilisons le mode d'évaluation prédictif séquentiel (Gama et al., 2009). Comme un classifieur évolutifs essaye d'abord de reconnaître un geste, puis s'en sert ensuite pour apprendre une fois qu'il dispose de sa vrai étiquette, nous évaluons nos systèmes d'une manière similaire. Ainsi, chaque donnée est d'abord utilisée comme donnée de test, puis ensuite comme donnée d'apprentissage. Les taux d'erreur sont ensuite calculés sur une fenêtre entre chaque point de test.

4.2 Performances en environnement non stationnaire

Nous évaluons ces différents systèmes sur un scénario non stationnaire simulant des changements de concept brusques. Pour simuler ces changements de concept, nous utilisons les 21 scripteurs du groupe 1 (gestes libres) à la suite. D'un scripteur à l'autre, les gestes d'une même classe sont différents ce qui oblige le classifieur à changer complètement son modèle.

Les taux d'erreur sont calculés sur les 56 derniers gestes (sur 173) de chaque scripteur pour évaluer les performances une fois le changement de concept appris. Les résultats (moyennés sur 20 ordres différents de présentation des scripteurs) sont présentés Figure 2.

Tout d'abord, les performances du système *Evolve* sans oubli sont sans équivoque. Sans oubli le système n'est pas capable à long terme de s'adapter aux changements de concepts. L'utilisation d'un facteur d'oubli exponentiel est indispensable, mais son choix est difficile.

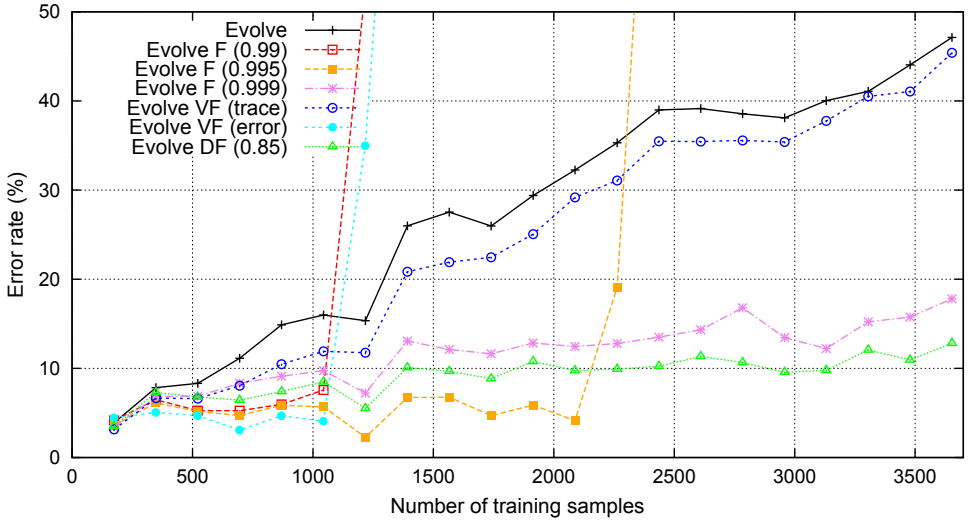

FIG. 2 – *Performances des différents systèmes en environnement non stationnaire.*

D'un côté, un facteur d'oubli de 0.99 ou 0.995 donne de bonnes performances au début, mais provoque l'explosion de la matrice de covariance et l'écroulement du système. De l'autre côté, un facteur d'oubli plus faible comme 0.999 est insuffisant et dégrade les performances du système (sans pour autant garantir que la matrice de covariance n'explosera pas à long terme).

Si l'utilisation d'un facteur d'oubli variable dans le temps peut donner de bons résultats sur des problèmes de régression, ce n'est pas du tout le cas dans notre contexte. L'utilisation de la trace de la matrice de covariance comme mesure d'information (*Evolve VF (trace)*) n'empêche pas certains éléments de la matrice de tendre vers zéro et le système perd très vite ses capacités d'adaptation. Au contraire, l'utilisation de l'erreur *a posteriori* (*Evolve VF (error)*) n'empêche pas l'explosion de la matrice de covariance lorsque le système est inégalement excité suivant les directions de l'espace.

Enfin, l'oubli directionnel est la seule approche stable, ce qui est logique car elle garantit que la matrice de covariance reste bornée. On observe cependant une légère augmentation du taux d'erreur tout au long de l'expérience. Cela vient du fait que certaines valeurs de la matrice de covariance se stabilisent à des valeurs relativement faibles, ce qui réduit légèrement les capacités d'adaptation du système.

5 Conclusion

Dans cet article nous avons étudié comment introduire de l'oubli dans l'apprentissage incrémental de classifieurs évolutifs basés sur des systèmes d'inférence floue (SIF). Plusieurs approches existent, principalement tirées du domaine de la régression, pour introduire de l'oubli dans l'algorithme des moindres carré récursifs utilisé pour l'apprentissage des SIF. Nous avons ainsi présenté l'utilisation d'un facteur d'oubli constant, d'un facteur d'oubli variable

(dans le temps), et enfin de l'oubli directionnel. Seul cette dernière technique garantit la stabilité du système à long terme et donne des résultats corrects (bien que non optimaux). Elle préserve le gain de l'algorithme sans provoquer l'explosion de la matrice de covariance, et permet au classifieur évolutif de s'adapter à un environnement non stationnaire.

Les bonnes performances de cette technique s'expliquent par le fait qu'elle garantit que la matrice de covariance utilisée dans l'algorithme des moindres carrés récursifs reste bornée. Cela suggère l'étude d'autres techniques pour introduire de l'oubli dans cet algorithme, en travaillant sur d'autres manières de maintenir la matrice de covariance bornée.

Ces travaux permettent ainsi d'envisager l'utilisation en-ligne « à vie »d'un classifieur évolutif sans dégradation de ses performances d'adaptation. Peu importe que l'utilisateur change, que les gestes d'une classe évoluent, ou que de nouvelles classes soit ajoutées, le classifieur va être capable de s'y adapter, même après une longue période d'utilisation stationnaire.

Références

Almaksour, A. et E. Anquetil (2011). Improving premise structure in evolving takagi-sugeno neuro-fuzzy classifiers. *Evolving Systems 2*, 25–33.

Angelov, P. et X. Zhou (2008). Evolving fuzzy-rule-based classifiers from data streams. *Fuzzy Systems, IEEE Transactions on 16*(6), 1462 –1475.

Bittanti, S., P. Bolzern, et M. Campi (1990). Exponential convergence of a modified directional forgetting identification algorithm. *Systems and Control Letters 14*(2), 131 – 137.

Cao, L. et H. Schwartz (2000). A directional forgetting algorithm based on the decomposition of the information matrix. *Automatica 36*(11), 1725–1731.

Delaye, A. et E. Anquetil (2013). HBF49 feature set : A first unified baseline for online symbol recognition. *Pattern Recognition 46*(1), 117–130.

Fortescue, T. R., L. S. Kershenbaum, et B. E. Ydstie (1981). Implementation of self-tuning regulators with variable forgetting factors. *Automatica 17*(6), 831–835.

Gama, J., R. Sebastião, et P. P. Rodrigues (2009). Issues in evaluation of stream learning algorithms. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, New York, NY, USA, pp. 329–338. ACM.

Hägglund, T. (1985). Recursive estimation of slowly time varying parameters. In *Preprints 7th IFAC*, York, UK, pp. 1137–1142.

Kulhavy, R. (1987). Restricted exponential forgetting in real-time identification. *Automatica 23*(5), 589–600.

Kulhavy, R. et M. B. Zarrop (1993). On a general concept of forgetting. *International Journal of Control 58*(4), 905–924.

Lughofer, E. et P. Angelov (2011). Handling drifts and shifts in on-line data streams with evolving fuzzy systems. *Applied Soft Computing 11*(2), 2057–2068.

Parkum, J., N. K. Poulsen, et J. Holst (1992). Recursive forgetting algorithms. *International Journal of Control 55*(1), 109–128.

Renau-Ferrer, N., P. Y. Li, A. Delaye, et E. Anquetil (2012). The ILGDB database of realistic pen-based gestural commands. *International Conference on Pattern Recognition (ICPR)*.

Tsukuba (Japan).

Salgado, M. E., G. C. Goodwin, et R. H. Middleton (1988). Modified least squares algorithm incorporating exponential resetting and forgetting. *International Journal of Control 47*(2), 477–491.

Summary

This paper study the use of forgetting for the incremental learning of evolving classifiers based on fuzzy inference systems. Several techniques are presented and compared on hand-written gesture recognition task in changing environement.