Table des matières

1 On peut répartir les différentes méthodologies en plusieurs catégories. 2

1.1 Méthodes utilisant un test statistique . . . 2

1.1.1 Concept drift detection based on Fisher's Exact test . . . 2

1.1.2 Detecting Concept Drift Using Statistical Testing . . . 2

1.1.3 Learning with Drift Detection . . . 3

1.1.4 Concept drift detection and adaptation with Hierarchical Hypothesis Testing . 3

1.1.5 Handling adversarial concept drift in streaming data . . . 4

1.2 Méthodes à base de modèles . . . 5

1.2.1 Tracking recurring contexts using ensemble classifiers : an application to email filtering . . . 5

1.2.2 Learning classification rules for telecom customer call data under concept drift 5

1.2.3 An Ensemble Approach for Incremental Learning in Nonstationary Environments 5

1.2.4 Data Stream Classification Guided by Clustering on Non Stationary Environments and Extreme Verification Latency . . . 6

1.2.5 A Novel Concept Drift Detection Method for Incremental Learning in Nonstationary Environmnents . . . 7

1.2.6 Concept Drift detection and adaptation in Big Imbalance industrial IoT data using ensemble learning method of offline classifiers . . . 7

1.2.7 Concept Drift Adaptation by exploiting historical knowledge . . . 8

1.2.8 Handling concept drift via model reuse . . . 8

1.2.9 Reacting to Different Types of Concept Drift : The Accuracy Updated Ensemble Algorithm . . . 9

1.3 Méthodes de détéction non supervisées . . . 10

1.3.1 An adaptive algorithm for anomaly and novelty detection in evolving data streams 10

1.3.2 Learning from Time-Changing Data with Adaptive Windowing . . . 10

1.4 Méthodes annexes . . . 11

1.4.1 Efficient Data Stream Classification via Probabilistic Adaptive Windows . . . 11

Chapitre 1

On peut répartir les différentes méthodologies en plusieurs catégories.

1.1 Méthodes utilisant un test statistique

1.1.1 Concept drift detection based on Fisher's Exact test

Danilo Rafael de Lima Cabral, Roberto Souto Maior de Barros, 2018

Mots clefs : Tests Statistiques, Détection sur score de modèle.

Résumé : L'article propose trois méthodes de détection de dérive à partir du test de Fischer fait sur les prédictions du modèle, les variantes sont en partie faites pour réduire le coût calculatoire. Les différences entres les méthodes sont les tests statistiques et la manière dont ils sont utilisés. Dans les trois cas, on suppose que l'on dispose des prédictions du modèle et de la vérité. On sépare les prédictions en 2 ensembles, les plus récentes et les anciennes, on va ensuite, à l'aide de tests statistiques regarder si une différence de distribution apparait conséquence d'une dérive.

FPDD utilise le test de Fischer quand le nombre d'erreurs où de prédicitions justes est faible (inférieur à 5) et utilise le test de l'hypothèse des proportion égale utilisé par la méthode STEPD le cas échéant. $T\left(r_0, r_r, n_o, n_r\right) = \frac{|r_0/n_o - r_r/n_r| - 0.5 \times (1/n_o + 1/n_r)}{\sqrt{\hat{p} \times (1-\hat{p}) \times (1/n_o + 1/n_r)}}$.

FSDD utilise le test de Fischer quand le nombre d'erreurs où de prédicitions justes est inférieure à 5 et utilise le test du chi deux le cas échéant.

FTDD utilise le test de Fischer exclusivement.

Les trois méthodes sont testées avec plusieurs jeux de données contre DDM, ECDD, SEED, FHDDM, STEPD et sortent avec en moyenne de meilleurs résultats.

À quelle(s) problématique(s) l'article répond ? Amélioration du temps de calcul et des performances de plusieurs méthodes

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? Il reste à explorer plus de tests statistiques et étudier une possible combinaison, càd l'utilisation de plusieurs tests en simultané et un méta modèle. Expérimenter l'impact de la taille des fenêtres d'observations prises en compte (elles sont brièvement étudiées).

1.1.2 Detecting Concept Drift Using Statistical Testing

Kyosuke Nishida and Koichiro Yamauch, 2007

Mots clefs Tests Statistiques, Détection sur score de modèle.

Résumé : L'article propose une méthode de détection de dérive à partir d'un test. On suppose que l'on dispose des prédictions du modèle et de la vérité. On sépare les prédictions en 2 ensembles, les plus récentes et les anciennes, on va ensuite, à l'aide de la statistique

$$T\left(r_0, r_r, n_o, n_r\right) = \frac{|r_0/n_o - r_r/n_r| - 0.5 \times (1/n_o + 1/n_r)}{\sqrt{\hat{p} \times (1-\hat{p}) \times (1/n_o + 1/n_r)}}$$

rejeter où accepter l'hypothèse de même distribution.

À quelle(s) problématique(s) l'article répond ? L'article propose une méthodologie de détection de dérive sur les scores des modèles.

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? La technique laisse à désirer en présence de dérive graduelle. Taille de la fenêtre.

1.1.3 Learning with Drift Detection

João Gama, Pedro Medas, Gladys Castillo, Pedro Rodrigues, 2004

Mots clefs

Résumé : Le but de la méthode développée ici est d'utiliser des intervalles de confiance de la loi normale appliqué aux erreurs faites par le modèle pour détecter une dérive. L'erreur devant logiquement diminuer au fur et à mesure que le modèle dispose de données d'entraînement faisant l'hypothèse que la distribution sous-jacente est la même, quand celle-ci augmente au delà d'un certain seuil, on est en présence d'une dérive.

À quelle(s) problématique(s) l'article répond ? L'article propose une méthodologie de détection de dérive sur les scores des modèles.

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? L'article ne compare pas les autres travaux, mais compare les différences de performances avec et sans la méthodologie.

1.1.4 Concept drift detection and adaptation with Hierarchical Hypothesis Testing

Shujian Yu, Zubin Abraham, Heng Wang ; 2019

Mots clefs Statistical Tests, Hierarchical, Adaptive Training

Résumé : Afin de détecter une dérive conceptuelle de manière stable, les auteurs mettent en place une méthodologie à deux niveaux. Un premier niveau détecte à travers l'instabilitée de 4 métriques relatant les performances du modèle, une dérive. Les auteurs en utilise 4 car l'utilisation d'une seule, comme dans d'autres méthodologies exisantes, ne permet pas l'obtention d'un délais de détection rapide et de précision forte.

Si une détection est relevée, alors un deuxième test est fait en comparant les performances du modèle entraîné sur les données d'avant la dérive et d'après. Les auteurs utilisent également une version adaptable de l'algorithme SVM permettant l'utilisation des données du concept précédent afin d'améliorer les performances du nouveau modèle, peu d'exemples d'entraînements étant disponibles après détéction d'un nouveau concept.

À quelle(s) problématique(s) l'article répond ? Les techniques développées permettent une détection plus rapide de la dérive. Le modèle adaptable permet de bonnes performances malgrès le nombre faible d'observations.

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? Rendre d'autres algorithmes Adaptables (Lire reférence 54).

1.1.5 Handling adversarial concept drift in streaming data

Tegjyot Singh Sethi , Mehmed Kantardzic, 2017

Mots clefs adversarial concept drift

Résumé : A lire.

À quelle(s) problématique(s) l'article répond ?

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ?

1.2 Méthodes à base de modèles

1.2.1 Tracking recurring contexts using ensemble classifiers : an application to email filtering

Ioannis Katakis, Grigorios Tsoumakas, Ioannis Vlahavas, 2009

Mots clefs

Résumé : On suppose que les données arrivent par séries où packets, on va les projeter les données dans un espace, puis, un algorithme non supervisé va les grouper par concepts, l'espace de projection permettant de comparer les instances avec la distance euclidienne. Chaque concept différent dispose de son modèle prédictif. Quand une nouvelle série d'observations arrive, elle est projeté dans l'espace, si elle ne correspond à aucun concept, on en créé un nouveau, sinon, on score.

La projection des variables se fait comme ceci : $z_i = \begin{cases} \left\{P_{i,j}^v : j = 1, \ldots, m, v \in V_i\right\}, & \text{si } f_i \text{ est nominale} \\ \{\mu_{i,j}, \sigma_{i,j} : j = 1, \ldots, m\}, & \text{si } f_i \text{ est numérique} \end{cases}$

Et où : $P_{i,j}^v = P\left(f_i = v \mid c_j\right), i \in [1, n], j \in [1, m], v \in V_i$. Où V_i est l'ensemble des valeurs possible que peut prendre la variable f_i.

À quelle(s) problématique(s) l'article répond ? À la projection des espaces des données dans un espace euclidien.

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? L'article fait l'hypothèse que des données d'un même batch appartiennent aux mêmes concepts.

1.2.2 Learning classification rules for telecom customer call data under concept drift

Black, M., Hickey, R, 2003

Mots clefs :

Résumé : L'article développe plusieurs versions d'une méthodologie basée sur les arbres de décisions. On suppose que les données arrivent par batchs. Un attribut de temporalité est introduit dans le jeu de données. Un arbre de décision est fait sur le dataset. Si l'attribut de temps est présent dans les décisions, alors on a un drift. Cela permet également de voir quelles variables ont été impactées par le drift. Seul les règle émanant des branches qui se trouvent dans la branche de l'arbre de décision où un attribut temporel est présent sont supprimées. Les règles qui n'ont pas été impactées restent.

À quelle(s) problématique(s) l'article répond ? Utiliser la temporalité afin de trouver les attributs en dépendant.

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? Utiliser cette méthode afin de vérifier si

1.2.3 An Ensemble Approach for Incremental Learning in Nonstationary Environments

Michael D. Muhlbaier and Robi Polikar, 2007

Mots clefs :

Résumé : L'article propose Learn^{++}.NSE NSE comme Non Stationnary Environment. On suppose que les données arrivent en par paquets, ou en batch. On suppose ici que les dernières données labélisées correspondent au concept le plus récent. Ainsi, à chaque fois qu'un batch de données labélisées arrive, on entraîne un nouveau modèle. On va ensuite regarder les taux d'erreurs respectifs de chaque ancien modèle sur l'ensemble des paquets de données.

On va ensuite attribuer une pondération à chaque modèle qui va dépendre des performances de ceux-ci sur les paquets de données. L'erreur des modèles sur les paquets récents va prendre plus de d'importance que les erreurs anciennes. Dans le cas d'une dérive récurrente, on va observer le poid d'un modèle osciller.

Le résultat de cette pondération va déterminer le poid de chacun des modèles dans la prédiction à venir.

À quelle(s) problématique(s) l'article répond ? L'article répond aux problèmes des dérives récurrentes.

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? L'article ne prend pas en compte la séparation des paquets en concepts. En effet, deux concepts peuvent arriver au sein d'un même paquet s'ils sont trop longs. S'il sont trop courts, alors les performances du modèle seront insatisfaisantes. La méthodologie utilise seulement les performances dans son discernement des concepts, un ajout de méthodes statistiques augmenterait sûrement la pertinence de la pondération. La méthodologie entraîne un modèle unique sans ce soucier des hyperparamètres.

1.2.4 Data Stream Classification Guided by Clustering on Non Stationary Environments and Extreme Verification Latency

VMA Souza, DF Silva, J Gama, GE Batista , 2015

Cité 62 fois, SIAM

Mots clefs : Semi-supervisé

Résumé : On dispose des labels seulement lors de l'initialisation et c'est un contexte où une dérive est présente. L'article présente une méthodologie semi-supervisée afin de prédire dans un contexte où il y a une dérive. Le contexte est un contexte de dérive incrémentale, le nombre de classes est supposé connu et ne change pas, on dispose initialement d'un nombre d'exemples labélisés.

À l'initialisation, on va attribuer chaque exemple à un cluster selon sa classe. On aura donc autant de clusters que de classes. Les centroïdes sont calculés.

On va ensuite récupérer des exemples non labélisés, utiliser k-means pour les répartir en clusters. Pour cela, lors de l'initialisation, on utilise les centroïdes des clusters précédement définis, par les classes, comme centroïdes d'initialisation. Une fois les nouveaux points répartis en clusters, on va attribuer aux instances non labélisées composant les clusters, le label du cluster de l'itération précédente qui est le plus proche. Les nouveaux exemples labelisés, on peut réentraîner le modèle sur les exemples les plus récents. On peut ainsi recommencer l'itération dès l'arrivé de nouveau exemples. On oscille ainsi entre classification supervisée et clustering.

À quelle(s) problématique(s) l'article répond ? L'article apporte une méthode pour traiter la dérive dans un contexte semi-supervisé. Dans un contexte où les labels des observations ne sont disponibles que lors de l'initialisation.

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? On peut imaginer que dans le cas d'une dérive de forte amplitude brusque, ou lors de dérives qui s'enchaînent sur une temporalité longue, le système perde en efficacité.

1.2.5 A Novel Concept Drift Detection Method for Incremental Learning in Nonstationary Environmnents

Z Yang, S Al-Dahidi, P Baraldi, E Zio, 2019

Cité 4 fois, ieeexplore.ieee.org

Mots clefs : Détection par analyse du modèle

Résumé : L'article détaille une méthode qui, pour détecter la dérive, va réentrainer un modèle (ELM) Extreme Learning Machines, qui est un modèle type réseau de neurones à une unique couche cachée. Ce type de modèle à la particularité qu'il peut être actualisé avec l'arrivée de nouvelles données quel que soit leur nombre.

Une fois le modèle entraîné, à l'arrivé de nouveaux exemples labélisés, un réentrainement s'opère. La différence de poid des neurones de la couche caché du modèle réentrainé avec les poids du modèle avant réentrainement est calculée. Les auteurs développent une distance faite pour mesurer l'écart entre les poids des deux modèles.

Si cette différence est supérieure à un seuil, dont la méthodologie de détermination est détaillé dans l'article. Un calcul, tiré de travaux récents pour quantifier le nombre d'exemples labélisés nécessaires à un réentraînement efficaces du point de vue des performances prédictives et en terme de coût caclulatoire.

À quelle(s) problématique(s) l'article répond ? Comment détecter une dérive avec le modèle sans regarder les performances de celui-ci ? Comment trouver le compromi idéal pour un savoir quand faire les réentraînements ?

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? Utiliser une méthodologie similaire sur d'autre algorithmes prédictif, calculer une distance entre les arbres de décisions, où trouver un seuil a partir duquel un changement des coefficients de la regression linéaire indiquerait une dérive.

1.2.6 Concept Drift detection and adaptation in Big Imbalance industrial IoT data using ensemble learning method of offline classifiers

Chun-Cheng Lin, Der-Jiunn Deng, 2019

Cité N fois, Revue

Mots clefs : Imbalanced Problems

Résumé : L'article propose une solution d'apprentissage automatique sur un problème contrebalancé en présence d'une dérive conceptuelle. Le traitement des données contrebalancées ce fait par la technique SMOTE, qui permet une création de points sythétiques de la classe minoritaire. La détection de la dérive se fait par une surveillance des performances du modèle par des seuils, on a affaire a une détection par seuil statistique de 4 métriques dérivés des prédictions du modèle et de la vérité.

À quelle(s) problématique(s) l'article répond ? Adresser un problème contrebalancé.

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? L'évaluation sur données réelles est faite, mais sans comparé à d'autres méthodologies.

1.2.7 Concept Drift Adaptation by exploiting historical knowledge

Yu Sun, Ke Tang, Zexuan Zhu, Xin Yao, 2018

Cité 49 fois, IEEE Explore

Mots clefs :

Résumé : L'article détaille une méthode : DTEL pour prédire en présence de dérive. La méthodologie conserve un pool de modèles (des arbres de décisions) de taille fixe. On suppose les données arrivant par batch D. A l'arrivé d'un batch labélisé à l'instant t :

- Un nouveau modèle est entrainé sur les données du batch D_t
- Les anciens modèles sont actualisés de la façon suivante : Pour chacun des anciens modèles, les données du nouveau batch sont répartis dans les feuilles de l'arbre et les probabilités des feuilles sont actualisés en conséquence. Les feuilles n'ayant pas atteint un critère de stop continuent d'être séparées avec les nouvelles données jusqu'à ce que le critère de stop prenne effet. Cela permet de garder la structure de l'arbre entraîné sur des données antérieures tout en prenant en compte le batch D_t.
- On ajoute le nouveau modèle au pool, et si la taille du pool excède un la taille maximum, on retire le modèle apportant le moins d'originalité selon la mesure : $\mathrm{div}(S) = 1 - \frac{1}{\sum_{1 \leq i \neq j \leq m} 1} \sum_{1 \leq i \neq j \leq m} Q(f_i, f_j)$ où $Q(f_i, f_j) = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}}$ tel que N^{ab} soit le nombre d'exemple tel que le modèle f_i prédise a et que le modèle f_j prédise b. Ici 1 représente une classification correcte, et 0 une erreur.

Les modifications avec les données du nouveau batch sur les anciens modèles sont temporaires et les modèles retrouvent leurs état d'origine avec l'arrivé d'un nouveau batch labélisé. Le poids qu'a chacun des modèles dans la prédiction finale est calculé en fonction des performances de ceux-ci sur le batch D_t

À quelle(s) problématique(s) l'article répond ?

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ?

1.2.8 Handling concept drift via model reuse

Peng Zhao, Le-Wen Cai, Zhi-Hua Zhou, 2019

Cité 6 fois, Springer

Mots clefs : Ensemble model

Résumé : On suppose les données arrivant de manière incrémentale et les labels arrivant peut après. La méthodologie utilise un pool de modèles linéraires. Un nouveau modèle est ajouté au pool quand une nouvelle dérive à été détectée où lorsque un nombre p d'exemples ont été vu depuis le dernier entraînement. Ici on utilise ADWIN comme detecteur de dérive.

La contribution de chacun des modèles est basé sur les perfomances de ceux si sur les derniers exemples labélisés. Lors de l'ajout d'un modèle ce dernier est entraîner en prenant compte des poids des modèles du pool.

$$\hat{\mathbf{w}}_k = \arg\min_{\mathbf{w}} \left\{ \frac{1}{m} \sum_{i=1}^{m} \ell\left(\langle \mathbf{w}, \mathbf{x}_i \rangle, y_i\right) + \mu\Omega\left(\mathbf{w}, \mathbf{w}_p\right) \right\}$$

où l est la fonction coût, x_i et y_i les données du dernier batch S_k. On a $\mathbf{w}_p = \sum_{j=1}^{k-1} \beta_j \hat{\mathbf{w}}_j$ où β_j est le poid du modèle h_j. Ω est la fonction de pénalisation.

Si le nombre de modèle est superieur de 1 à la taille du pool, on supprime le plus ancien.

À quelle(s) problématique(s) l'article répond ?

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? Faire un algorithme ayant de bonne performance quand on a une abscence de y immédiate. Tel que les perfomances ne se dégradent pas trop, même si un drift à été détecté et qu'aucun exemple labélisé on été reçu. Avec une double mesure.

1.2.9 Reacting to Different Types of Concept Drift : The Accuracy Updated Ensemble Algorithm

Dariusz Brzezinski and Jerzy Stefanowski, 2014

Cité N fois,

Mots clefs : Ensemble model,

Résumé : Les auteurs proposent dans leurs approche, d'utiliser un pool de modèle, les modèles ici sont des arbres de décision, mais tous les algorithmes pouvant être réentrainés en ligne peuvent être choisi. Avec l'arrivée d'un nouveau batch, un nouveau modèle est entraîné. Les poids que prennent chaque algorithme du pool est calculé en fonction des performances obtenues sur le dernier batch.

Les anciens modèles sont actualisés en prenant en compte le nouveau batch. Si la taille en mémoire des modèles dépasse un seuil, alors les modèles sont élagués.

À quelle(s) problématique(s) l'article répond ? L'article apporte une réponse avec à la question de taille de batch. Cette méthodologie peut s'accomoder de tout petit batch.

1.3 Méthodes de détéction non supervisées

1.3.1 An adaptive algorithm for anomaly and novelty detection in evolving data streams

Mohamed-Rafik Bouguelia, Slawomir Nowaczyk, Amir H. Payberah ; 2018

Mots clefs : Gas Network Generator, Self Organising Maps

Résumé : L'article propose une amélioration du modèle GNG, en apportant les modifications nécessaires pour une meilleure adaptabilité à un changement de distribution.

L'algorithme des GNG est un algorithme non supervisé *online*. On initialise un nombre n de clusters ou de sommets de même dimension que le jeu de données. À chaque apparition d'un exemple x, on calcul les deux sommets les plus proches. On va décaler le sommet le plus proche légèrement en direction du nouveau point, puis on trace une arrête entre les deux sommets les plus proches d'âge 0. On incrémente de 1 toutes les arrêtes, puis, on supprime les arrêtes d'âge supérieur à un seuil. Puis, on va supprimer tous les sommets isolés. Toutes les λ itérations on va créer un nouveau sommet à partir du sommet existant ayant la distance moyenne entre ses points x et le centre la plus élevées.

Le problème de ce modèle est qu'un grand nombre de paramètres est nécessaire à son fonctionnement et donc que l'on manque d'adaptabilitée, que certains vieux neurones ne sont pas supprimés alors que peu utiles.

L'algorithme proposé le GNG-A comme adaptive, l'adaptabilité signifie que l'initialisation des paramètres perd son importance les hyperparamètres évoluant. Le processus de suppression et création des sommets n'est plus systématique, mais prend en compte la dérive.

À quelle(s) problématique(s) l'article répond ? La paramétrisation des hyperparamètres n'est plus capitale, les sommets ne sont plus créés et suprimés de façon systématique. L'algorithme est maintenant utilisable complètement en ligne.

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? On a ici un algorithme non supervisé. Adapter ces méthodes à un problème supervisé. Adapter au concepts récurrents lors de la suppression en gardant en mémoire les anciens sommets. Ne traite pas les variables n'étant pas de type numériques.

1.3.2 Learning from Time-Changing Data with Adaptive Windowing

A Bifet, R Gavalda , 2007

Mots clefs : ADWIN

Résumé : Un algorithme de sélection de fenêtre d'apprentissage est présenté. L'algorithme conserve les n plus récentes observations dans une fenêtre W de taille n avant de la séparer en deux parties W_1 et W_2 de taille n_1 et n_2 tel que $n_1 + n_2 = n$.

Avec $m = \frac{1}{1/n_0+1/n_1}$ (moyenne harmonique de n_0 et n_1) ; $\delta' = \frac{\delta}{n}$ et $\epsilon_{cut} = \sqrt{\frac{1}{2m} \cdot \ln \frac{4}{\delta'}}$.

Tant que l'on observe pas $|\hat{\mu}_{W_0} - \hat{\mu}_{W_1}| \geq \epsilon_{cut}$ où $\hat{\mu}_{W_0}$ est la moyenne de la fenêtre W_0 et $\hat{\mu}_{W_1}$ celle de W_1, on supprime les observations les plus anciennes

À quelle(s) problématique(s) l'article répond ? Détection d'une dérive, sélection des observations correspondant au concept le plus récent.

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ? Utiliser ADWIN pour séparer les données en concepts différents $C_1, C_2, ...C_i$ avec des statistiques telles que la moyenne, la variance, les quantiles... Cela permettrait, quand un nouveau concept apparaîtrait de vérifier si l'on ne dispose pas d'autres instances de ce concept dans le cas d'un concept récurrent.

Implémentation : `https://scikit-multiflow.readthedocs.io/en/stable/api/generated/skmultiflow.drift_detection.ADWIN.html#skmultiflow.drift_detection.ADWIN`

1.4 Méthodes annexes

1.4.1 Efficient Data Stream Classification via Probabilistic Adaptive Windows

Albert Bifet, Jesse Read, Bernhard Pfahringer, Geoff Holmes ; 2013

Mots clefs : Memory optimisation, online learning

Résumé : L'article détaille une méthode : DTEL pour prédire en présence de dérive. La méthodologie conserve un pool de modèles (des arbres de décisions) de taille fixe. On suppose les données arrivant par batch D. A l'arrivé d'un batch labélisé à l'instant t :

- Un nouveau modèle est entrainé sur les données du batch D_t
- Les anciens modèles sont actualisés de la façon suivante. Pour chacun des anciens modèles, les données du nouveau batch sont répartis dans les feuilles de l'arbre et les probabilités des feuilles sont actualisés en conséquence. Les feuilles n'ayant pas atteint un critère de stop continuent d'être séparées avec les nouvelles données jusqu'à ce que le critère de stop prenne effet. Cela permet de garder la structure de l'arbre entraîné sur des données antérieures tout en prenant en compte le batch D_t.
- On ajoute le nouveau modèle au pool, et si la taille du pool excède un la taille maximum, on retire le modèle apportant le moins d'originalité selon la mesure : $\text{div}(S) = 1 - \frac{1}{\sum_{1 \leq i \neq j \leq m} 1} \sum_{1 \leq i \neq j \leq m} Q(f_i, f_j)$ où $Q(f_i, f_j) = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}}$ tel que N^{ab} soit le nombre d'exemple tel que le modèle f_i prédise a et que le modèle f_j prédise b. Ici 1 représente une classification correcte, et 0 une erreur.

Les modifications avec les données du nouveau batch sur les anciens modèles sont temporaires et les modèles retrouvent leurs état d'origine avec l'arrivé d'un nouveau batch labélisé. Le poids qu'a chacun des modèles dans la prédiction finale est calculé en fonction des performances de ceux-ci sur le batch D_t

À quelle(s) problématique(s) l'article répond ?

Quelles sont les pistes à explorer et ont-elles été explorées par d'autres articles ?