CRAFTML, une forêt aléatoire efficace pour l'apprentissage multi-label extrême

Wissam Siblini, Frank Meyer, Pascale Kuntz

CRAFTML, une forêt aléatoire efficace pour l'apprentissage multi-label extrême

Wissam Siblini*,**, Frank Meyer *
Pascale Kuntz**

*Orange Labs - Lannion, France
prenom.nom@orange.com,
**Laboratoire des Sciences du Numérique de Nantes (LS2N) - Nantes, France
prenom.nom@univ-nantes.fr

Résumé. L'apprentissage multi-label extrême (noté XML pour "eXtreme Multi-label Learning") considère de grands volumes de données où chaque observation est annotée avec quelques labels parmi des centaines de milliers de possibilités. Les méthodes basées sur les arbres, qui divisent hiérarchiquement l'apprentissage en sous-problèmes à petite échelle, sont particulièrement prometteuses dans ce contexte pour réduire les complexités d'apprentissage et de prédiction et pour ouvrir la voie à la parallélisation. Cependant, les meilleures approches actuelles n'exploitent pas la diversification des arbres qui a pourtant montré son efficacité dans les forêts aléatoires et elles ont recours à des stratégies de partitionnement complexes. Pour surmonter ces limites, nous introduisons ici un nouvel algorithme de forêt avec des arbres diversifiés et une stratégie de partitionnement adaptée à l'XML appelé CRAFTML. Des comparaisons expérimentales sur huit jeux de données tirés de la littérature extrême montrent qu'il est plus performant que les autres approches arborescentes de l'état de l'art.

1 Introduction

La classification multi-label a reçu une attention considérable au cours de la dernière décennie et récemment, stimulée par des applications impliquant de grands ensembles de données, elle a été étendue à des problèmes où le nombre de labels peut dépasser le million (Agrawal et al., 2013). Dans ce nouveau contexte, appelé apprentissage multi-label extrême (noté XML en anglais pour eXtreme Multi-Label Learning) où les données sont très creuses et ont un très grand nombre de dimensions, les algorithmes classiques ne parviennent pas à passer à l'échelle ou voient leurs performances prédictives se dégrader. Pour tenter de surmonter ces difficultés, la recherche s'est récemment orientée vers trois directions : (i) utiliser des astuces d'optimisation et la parallélisation sur des "superordinateurs" (Yen et al., 2017), (ii) réduire la dimension des données pour obtenir un problème latent soluble avec des approches multi-label classiques (Bhatia et al., 2015) ou (iii) partitionner hiérarchiquement le problème initial en sous-problèmes à petite échelle (Prabhu et Varma, 2014). La décomposition arborescente de la troisième stratégie présente plusieurs avantages. En découpant l'apprentissage en sous-tâches,

elle réduit la complexité de l'entraînement et des prédictions et ouvre la voie à la parallélisation. Et, sa séquence de décisions successives permet une grande expressivité.

Motivés par ces propriétés, nous présentons une nouvelle approche arborescente rapide et précise appelée CRAFTML (Clustering-based RAndom Forest of predictive Trees for extreme Multi-label Learning). Comme FastXML (Prabhu et Varma, 2014) qui fait partie des meilleures approches arborescentes pour l'apprentissage multi-label extrême, CRAFTML est une forêt d'arbres de décision supervisés où les conditions de séparation des instances à chaque nœud sont multivariées. Mais CRAFTML a deux différences de fond avec FastXML : (i) il implémente de la diversité entre les arbres avec une stratégie qui s'approche de celle des "forêts aléatoires" mais qui s'en distingue non seulement car il remplace les sélections aléatoires par des projections aléatoires pour préserver plus d'informations mais aussi car il les applique aux labels en plus des attributs ; (ii) il utilise une nouvelle stratégie de séparation qui est plus simple et qui a une très faible complexité. Des expériences numériques sur huit jeux de données de la littérature multi-label extrême montrent que CRAFTML dépasse les approches arborescentes XML avec un temps d'apprentissage inférieur et une consommation de mémoire plus faible. Il est également compétitif avec les autres approches de l'état de l'art XML.

Le reste de cet article de résumé est organisé comme suit. La section 2 rappelle brièvement les travaux récents dans le domaine de l'apprentissage multi-label extrême. La section 3 décrit notre nouvelle proposition CRAFTML. La section 4 compare les performances de CRAFTML avec l'état de l'art XML. Les références de toutes les méthodes citées ici sont disponibles dans l'article original (Siblini et al., 2018).

2 Travaux connexes

En raison de l'intérêt croissant de l'apprentissage multi-label au cours de la dernière décennie, de nombreux algorithmes ont été proposés (Zhang et Zhou, 2014). Plusieurs expérimentations numériques ont mis en évidence les bonnes performances de la forêt aléatoire multi-label RF-PCT et de la méthode hiérarchique HOMER. Cependant, ces algorithmes ne sont pas adaptés aux dimensions (10^5 à 10^7) de l'apprentissage multi-label extrême. Trois stratégies différentes sont aujourd'hui développées pour résoudre le problème de passage à l'échelle : les astuces d'optimisation et de parallélisation, la réduction de dimension et la décomposition hiérarchique arborescente.

Dans ce résumé, nous nous concentrons particulièrement sur les méthodes arborescentes (LPSR, FastXML, son extension PFastReXML et PLT). De façon générale, celles-ci transforment le problème initial à grande échelle en une série de sous-problèmes à petite échelle en partitionnant de manière hiérarchique l'ensemble des instances ou l'ensemble des labels. Ces différents sous-ensembles sont associés aux nœuds d'un arbre. L'ensemble initial associé à la racine est partitionné en k sous-ensembles qui sont eux-mêmes associés aux k nœuds enfants de la racine. Le processus de décomposition est répété jusqu'à ce qu'une condition d'arrêt soit vérifiée sur les sous-ensembles. Dans chaque nœud, deux problèmes d'optimisation sont soulevés : (i) construire une partition pour optimiser un critère donné, et (ii) définir une condition ou construire un classifieur pour décider, à partir des attributs d'une instance donnée, du sous-ensemble de la partition auquel l'associer. Dans la phase de prédiction, une nouvelle instance suit un chemin de la racine jusqu'à une feuille (arbre d'instances) ou plusieurs feuilles (arbre de labels) déterminées par les décisions locales successives dans les nœuds puis, des

prédictions en sont déduites. Pour un arbre de labels, les labels associés aux feuilles atteintes sont celles prédites avec une probabilité non nulle. Pour un arbre d'instances, la prédiction est donnée par un classifieur local appris uniquement à partir des instances de la feuille atteinte.

Dans la littérature XML, trois approches arborescentes récentes ont été proposées : LPSR et FastXML respectivement basées sur un seul arbre d'instances k-aire et une forêt d'arbres d'instances binaires ($k = 2$), et PLT basé sur un arbre de labels. LPSR vise à regrouper dans un même sous-ensemble les instances qui partagent des attributs et des labels communs. FastXML vise à minimiser une fonction de perte basée sur le "nDCG" (normalized Discounted Cumulative Gain) qui tend à regrouper des instances avec des labels communs. FastXML a été récemment étendu à une nouvelle version appelée PFastReXML afin de mieux prendre en compte la distribution à longue traîne des labels. PLT construit un arbre de labels en séparant récursivement les labels en sous-ensembles pour regrouper ceux qui co-occurrent dans les mêmes instances. La récursion s'arrête lorsque les sous-ensembles terminaux contiennent un seul label. Un classifieur multi-label est ensuite appris à chaque nœud pour permettre d'estimer des probabilités de suivre les différents chemins racine-feuille de l'arbre conditionnellement aux attributs. L'apprentissage est tel que les chemins se terminant sur les feuilles associées aux labels pertinents ont une forte probabilité.

Pour aller vers des stratégies de séparation moins complexes que l'état de l'art et pour intégrer une diversité entre les arbres adaptée à l'apprentissage extrême, nous introduisons une nouvelle approche arborescente appelée CRAFTML.

3 CRAFTML

CRAFTML construit une forêt d'arbres d'instances k-aires en suivant le schéma commun des méthodes basées sur les arbres d'instances rappelé dans la section 2. Sa stratégie de partitionnement de nœuds consiste à regrouper les instances avec des labels communs dans les mêmes nœuds/feuilles en visant à satisfaire deux contraintes : chaque arbre/nœud doit apprendre sur des instances projetées aléatoirement pour assurer la diversité et le processus de partitionnement doit effectuer des opérations de faible complexité pour assurer le passage à l'échelle. Par conséquent, l'apprentissage d'un nœud dans CRAFTML suit les trois étapes ci-dessous :

1. Projeter aléatoirement, dans des espaces de dimension réduite, les attributs et les labels des instances du nœud. Ces projections, dont les coefficients sont générés sur demande avec une graine et un "hash", sont orthogonales et creuses de type hashing trick qui est empiriquement efficace sur les données XML (Siblini et al., 2018).
2. Partitionner les instances en k sous-ensembles temporaires en appliquant l'algorithme des k-moyennes sphériques (algorithme de Loyd) sur la matrice des labels projetés. La métrique du cosinus est utilisée car elle est rapide et adaptée aux données creuses. Les centroïdes des clusters sont initialisés avec la stratégie k-means ++ qui améliore la stabilité et les performances de l'algorithme par rapport à une initialisation aléatoire.
3. Apprendre un classifieur multi-classe très simple (plus proche centroïde dans nos expériences) pour associer chaque instance à son sous-ensemble temporaire pertinent (c'est-à-dire l'indice de cluster calculé à l'étape 2) à partir de son vecteur d'attributs

projetés. Puis, partitionner les instances en k sous-ensembles finaux (nœuds enfants) selon la classe que leur associe le classifieur.

Les étapes 2 et 3 sont accélérées avec une stratégie d'échantillonnage sans remise. Une fois qu'un arbre a été formé, chaque feuille stocke un vecteur correspondant à la moyenne des vecteurs de labels de ses instances. Dans la phase de prédiction, pour chaque arbre, l'instance d'entrée suit un chemin racine-feuille déterminé par les décisions successives des classifieurs des nœuds parcourus et la prédiction fournie est le vecteur stocké dans la feuille atteinte. La forêt agrège les prédictions des arbres avec une simple moyenne.

	Arbres de labels		Arbres d'instances			
	HOMER	PLT	RF-PCT	LPSR	FastXML	CRAFTML
Adapté à l'XML	Non	Oui	Non	Oui	Oui	Oui
Projections des attributs	Non	Oui*	Oui**	Non	Non	Oui
Projections des labels	Non	Non	Non	Non	Non	Oui
Plusieurs arbres	Non	Non	Oui	Non	Oui	Oui
Arbres binaires	Non	Non	Oui	Non	Oui	Non
Condition de séparation multivariée	Oui	Oui	Non	Oui	Oui	Oui

TAB. 1 – *Comparaison des caractéristiques des méthodes arborescentes. *depend de la taille mémoire, **selection aléatoire d'attributs.*

Les principales similitudes et différences entre CRAFTML et les autres approches arborescentes de l'état de l'art sont résumées dans le tableau 1. Grâce aux projections aléatoires, à la simplicité du séparateur, et à l'exploitation du caractère creux des données, les complexités spatiales et temporelles de CRAFTML sont faibles et indépendantes du nombre d'attributs et de labels (Siblini et al., 2018) ce qui permet à l'approche de passer à l'échelle en XML.

4 Comparaisons expérimentales

Nous comparons CRAFTML avec les meilleures méthodes arborescentes de l'état de l'art (FastXML, PFastReXML, LPSR, PLT) présentées dans la section 2 sur huit ensembles de données apprentissage/test de référence en XML détaillés dans le repository [1]. Un hyperparamètrage standard a été determiné pour CRAFTML et a été fixé pour l'intégralité des expérimentations (Siblini et al., 2018). Il assure une comparaison équitable avec FastXML et son extension. Le nombre de labels d_y, le nombre d'attributs d_x, le nombre d'instances d'apprentissage n et le nombre d'instances de test n_S des jeux de données sont rappelés dans le tableau 2.

CRAFTML a de meilleures performances prédictives que les autres approches dans la plupart des cas. Pour WikiLSHTC-325K et Amazon-670K, la domination de PFastreXML s'explique en partie par le fait que ce dernier est entraîné avec les propensions des labels calculées avec des informations externes supplémentaires (hiérarchie des labels de Wikipedia et Amazon). Les comparaisons en temps de calcul et en consommation mémoire ne sont pas

1. `http://manikvarma.org/downloads/XC/XMLRepository.html`

	Bibtex $d_x = 1836, d_y = 159$ $n = 4880, n_{\mathcal{S}} = 2515$			Delicious $d_x = 500, d_y = 983$ $n = 12920, n_{\mathcal{S}} = 3185$			EURLex-4K $d_x = 5000, d_y = 3993$ $n = 15539, n_{\mathcal{S}} = 3809$			Wiki10-31K $d_x = 101938, d_y = 30938$ $n = 14146, n_{\mathcal{S}} = 6616$		
	P@1	P@3	P@5	P@1	P@3	P@5	P@1	P@3	P@5	P@1	P@3	P@5
CRAFTML	**65.15**	**39.83**	28.99	**70.26**	63.98	59.00	**78.81**	**65.21**	**53.71**	**85.19**	**73.17**	**63.27**
PFastReXML	63.46	39.22	**29.14**	67.13	62.33	58.62	75.45	62.70	52.51	83.57	68.61	59.10
FastXML	63.42	39.23	28.86	69.61	**64.12**	**59.27**	71.36	59.90	50.39	83.03	67.47	57.76
LPSR	62.11	36.65	26.53	65.01	58.96	53.49	76.37	63.36	52.03	72.72	58.51	49.50
PLT	-	-	-	-	-	-	-	-	-	84.34	72.34	62.72
	WikiLSHTC-325K $d_x = 1617899, d_y = 325056$ $n = 1778351, n_{\mathcal{S}} = 587084$			Delicious-200K $d_x = 782585, d_y = 205443$ $n = 196606, n_{\mathcal{S}} = 100095$			Amazon-670K $d_x = 135909, d_y = 670091$ $n = 490449, n_{\mathcal{S}} = 153025$			AmazonCat-13K $d_x = 203882, d_y = 13330$ $n = 1186239, n_{\mathcal{S}} = 306782$		
	P@1	P@3	P@5	P@1	P@3	P@5	P@1	P@3	P@5	P@1	P@3	P@5
CRAFTML	**56.57**	34.73	25.03	**47.87**	**41.28**	**38.01**	37.35	33.31	30.62	92.78	**78.48**	63.58
PFastReXML	56.05	**36.79**	**27.09**	41.72	37.83	35.58	**39.46**	**35.81**	**33.05**	91.75	77.97	**63.68**
FastXML	49.75	33.10	24.45	43.07	38.66	36.19	36.99	33.28	30.53	**93.11**	78.20	63.41
LPSR	27.44	16.23	11.77	18.59	15.43	14.07	28.65	24.88	22.37	-	-	-
PLT	45.67	29.13	21.95	45.37	38.94	35.88	36.65	32.12	28.85	91.47	75.84	61.02

TAB. 2 – *Comparaison entre CRAFTML et l'état de l'art XML. Le meilleur résultat est en gras. Les mesures, correspondant aux précisions à 1, 3 et 5 sont celles classiquement utilisées en apprentissage multi-label extrême.*

détaillées ici mais montrent que CRAFTML est très compétitif (Siblini et al., 2018). Comparé aux autres méthodes, son temps d'entraînement observé est inférieur en moyenne et la taille de son modèle est plus petite. Ces mesures sont cohérentes avec les résultats théoriques : en raison de la stratégie d'échantillonnage et de la réduction de la dimension résultant des projections aléatoires, les complexités de CRAFTML sont les plus faibles. Son temps de prédiction est cependant plus élevé que les autres même si sa complexité est équivalente.

Nous avons également comparé CRAFTML aux autres approches non parallélisées de l'état de l'art XML (SLEEC, PDSparse, AnnexML) et les performances prédictives sont comparables mais CRAFTML est plus rapide et/ou consomme moins de mémoire (Siblini et al., 2018). En comparaison avec les méthodes conçues pour la parallélisation et nécessitant des ordinateurs avec des centaines de coeurs (PPDSparse, DISMEC) les conclusions sont mixtes et dépendent du jeu de données. Les jeux WikiLSHTC-325K et Amazon-670K semblent favoriser les deux approches parallèles par rapport à toutes les méthodes arborescentes. Cependant, en termes de complexités, CRAFTML est beaucoup plus intéressant que DISMEC et PPDSparse.

5 Conclusion

Notre nouvelle méthode d'apprentissage multi-label extrême CRAFTML est compétitive avec les méthodes de l'état de l'art extrême. Et, contrairement à ces dernières, elle ne s'appuie pas sur un schéma d'optimisation complexe. Elle combine des blocs d'apprentissage simples et rapides (par exemple un clustering avec k-means, un classifieur multi-classe très naïf), ce qui permet d'envisager des extensions et d'atteindre les performances requises par les défis sociétaux et techniques actuels. Avec la dimensionnalité croissante des données, l'apprentissage automatique recourt de plus en plus aux supercalculateurs. Mais cet accès est loin d'être disponible partout aujourd'hui et son coût va fixer des limites à l'avenir. Par conséquent, (i) des algorithmes d'apprentissage machine économes en ressources et évolutifs sont nécessaires pour favoriser la démocratisation des nombreuses applications du monde réel qui dépendent encore

du calcul standard. En contraste, le cloud computing (Hashem et al., 2015) et le développement croissant des supercalculateurs (Dean et al., 2018) nécessitent également (ii) des méthodes qui exploitent pleinement les ressources de calcul disponibles en étant, en particulier, facilement parallélisables. CRAFTML s'inscrit dans les deux cadres (i) et (ii).

Références

Agrawal, R., A. Gupta, Y. Prabhu, et M. Varma (2013). Multi-label learning with millions of labels : Recommending advertiser bid phrases for web pages. In *Proceedings of the 22nd international conference on World Wide Web*, pp. 13–24. ACM.

Bhatia, K., H. Jain, P. Kar, M. Varma, et P. Jain (2015). Sparse local embeddings for extreme multi-label classification. In *Advances in neural information processing systems*, pp. 730–738.

Dean, J., D. Patterson, et C. Young (2018). A new golden age in computer architecture : Empowering the machine learning revolution. *IEEE Micro*.

Hashem, I. A. T., I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, et S. U. Khan (2015). The rise of "big data" on cloud computing : Review and open research issues. *Information Systems 47*, 98–115.

Prabhu, Y. et M. Varma (2014). Fastxml : A fast, accurate and stable tree-classifier for extreme multi-label learning. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 263–272. ACM.

Siblini, W., F. Meyer, et P. Kuntz (2018). Craftml, an efficient clustering-based random forest for extreme multi-label learning. In *International Conference on Machine Learning*, pp. 4671–4680.

Yen, I. E., X. Huang, W. Dai, P. Ravikumar, I. Dhillon, et E. Xing (2017). Ppdsparse : A parallel primal-dual sparse method for extreme classification. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 545–553. ACM.

Zhang, M.-L. et Z.-H. Zhou (2014). A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering 26*(8), 1819–1837.

Summary

eXtreme Multi-label Learning (XML) considers large number of instances annotated with a few labels among hundreds of thousands of possibilities. Tree-based methods, which hierarchically divide learning into small-scale subproblems, are particularly promising in this context to reduce the complexities of learning and predictions and to open the way to parallelization. However, the current best approaches do not exploit tree randomization which has yet shown its efficiency in random forests and they resort to complex partitioning strategies. To overcome these limitations, we introduce here a new forest algorithm with diversified trees and a partitioning strategy adapted to XML called CRAFTML. Experimental comparisons on eight XML datasets show that our approach is faster and more accurate than the other state-of-the-art tree-based methods.