# Etude comparative des méthodes pour la vérification des systèmes cyber-physiques basés machine learning

Arthur Clavière[1], Laura Altieri-Sambartolomé[1], Eric Asselin[1], Christophe Garion[2] et Claire Pagetti[3]

[1]Collins Aerospace [2]ISAE-SUPAERO [3]ONERA

Cet article est un résumé étendu de l'article "Verification of machine learning based cyber-physical systems: a comparative study", accepté à HSCC 2022 (25th ACM International Conference on Hybrid Systems: Computation and Control).

**Contexte** Malgré un fort potentiel, les algorithmes issus d'apprentissage automatique *e.g.,* les réseaux de neurones, demeurent inutilisés dans les systèmes embarqués critiques. Le principal frein à cette utilisation réside dans la difficulté à certifier ces algorithmes. Cette difficulté est essentiellement liée au problème de la spécification du comportement attendu pour ces algorithmes. En effet, si une telle spécification pouvait être obtenue facilement, l'apprentissage ne présenterait plus aucun intérêt : la spécification serait suffisante et l'apprentissage ne serait plus nécessaire. Comment alors vérifier un réseau de neurones vis-à-vis d'une spécification qui n'existe pas, ou tout du moins n'est pas complète ?

Récemment, plusieurs travaux ont proposé une approche *système* permettant de contourner le problème de la spécification des réseaux de neurones en formulant le problème de vérification non pas au niveau du réseau de neurones mais au niveau du système qui l'utilise. Cet article s'intéresse à cette approche, en considérant une classe de système particulière : un système cyber-physique (CPS) où le contrôleur est un *classificateur* basé sur plusieurs réseaux de neurones. Ce type de système combine une partie physique en temps continu avec un contrôleur en temps discret, qui produit une commande parmi un ensemble fini à chaque exécution. Afin de choisir parmi cet ensemble fini de commandes, le contrôleur dispose d'une collection de réseaux de neurones. Un seul de ces réseaux est exécuté à chaque fois : le choix du réseau à exécuter est fonction de la commande précédente. L'étude de ce type de système est pertinente puisque plusieurs cas d'usage décrits dans la littérature entrent dans ce cadre, notamment dans le domaine aéronautique [5]. Le fait de choisir parmi plusieurs réseaux en fonction de l'état du contrôleur (*i.e.,* la commande précédente) permet d'avoir des réseaux plus petits et qui s'exécutent plus rapidement, chose importante dans un contexte embarqué.

Différents travaux se sont intéressés à la vérification d'un tel système. Dans cet article, nous proposons une revue détaillée de ces différentes méthodes, via les contributions suivantes :

**Modélisation** Un modèle est proposé pour le système d'intérêt, s'appuyant sur le formalisme des automates hybrides. Ce modèle prend en compte des hypothèses réalistes, notamment l'exécution non instantanée du contrôleur et des réseaux de neurones associés. Le problème de vérification est formulé comme un problème d'atteignabilité : montrer que l'ensemble des états atteignables du système est disjoint de l'ensemble des états d'erreurs.

**Description des méthodes de vérification** En s'appuyant sur le modèle proposé, les méthodes de vérification existantes sont présentées et les hypothèses sous-jacentes sont mises en évidence. Parmi ces méthodes, une approche par programmation linéaire en nombres entiers à été proposée et implémentée dans l'outil VENMAS [1]. Cette approche présente l'avantage d'offrir une représentation exacte du problème mais s'applique uniquement au cas où la dynamique du système est linéaire. Une autre approche, disponible dans l'outil NNV [6], propose de calculer une sur-approximation des états atteignables par le système en s'appuyant sur plusieurs domaines abstraits, à savoir les zonotopes et les *star sets*. Parallèlement au développement de cette dernière méthode, une approche similaire a été développée, implémentée dans l'outil SAMBA [3]. Celui-ci construit une sur-approximation qui s'avère être moins précise (mais potentiellement suffisante) et propose aussi plusieurs heuristiques pour accélérer la résolution du problème de vérification. Cette approche s'appuie d'une part sur des techniques de simulation garantie [2] pour évaluer la dynamique continue du système, et d'autre part des techniques d'interprétation abstraite dédiées à l'analyse de réseaux de neurones [7, 8, 4].

**Etude expérimentale** Afin de comparer expérimentalement les méthodes susmentionnées, nous avons considéré un ensemble de cas d'étude incluant deux systèmes d'anti-collision aéronautiques (VCAS et ACAS Xu), correspondant à des systèmes réalistes et représentatifs des potentielles futures utilisations des réseaux de neurones dans le domaine de l'aviation, ainsi qu'un pendule inversé placé sur un chariot en mouvement (Cartpole). Ces trois cas d'étude présentent différentes caractéristiques, pouvant influencer la performance de la vérification : différents types de dynamique, un nombre plus ou moins élevé de réseaux de neurones, eux-mêmes de taille plus ou moins grande. Pour chacun de ces cas d'étude, nous avons considéré un ensemble de problèmes de vérification, de difficulté plus ou moins élevée, afin d'évaluer le compromis entre *précision* et *passage à l'échelle* offert par chaque méthode de vérification. Ces problèmes de vérification ont été construit en considérant (1) un état initial pour le système, (2) une incertitude sur cet état et (3) un horizon temporel donné pour l'évaluation de la propriété d'intérêt. La difficulté du

problème de vérification est déterminée par (a) sa *criticité* : l'état initial choisi peut conduire, ou non, le système dans un état indésirable sans action du contrôleur, (b) sa *nature* : l'incertitude considérée peut concernée l'ensemble des variables d'états ou uniquement certaines d'entre-elles, et (c) son *horizon temporel*, plus ou moins grand.

Un extrait des résultats est donné en figure 1, où, pour chaque cas d'étude, on donne le pourcentage de problèmes qui ont pu être vérifiés par l'outil, ainsi que le temps cumulé nécessaire à la résolution de ces problèmes. Le meilleur outil est donc situé en haut à gauche sur ces diagrammes. Ces résultats mettent en évidence la moins bonne performance de VENMAS qui, du fait de sa plus grande précision, passe mal à l'échelle sur les problèmes difficiles du VCAS (grands horizons temporels), en plus de n'être applicable ni sur le ACAS Xu ni sur le Cartpole. Par ailleurs, les performances relatives de NNV et SAMBA dépendent du cas d'étude considéré : par exemple, sur le VCAS, NNV offre une approximation plus précise, précision déterminante pour éviter une explosion combinatoire du nombre de chemins explorés, d'où sa meilleure performance.

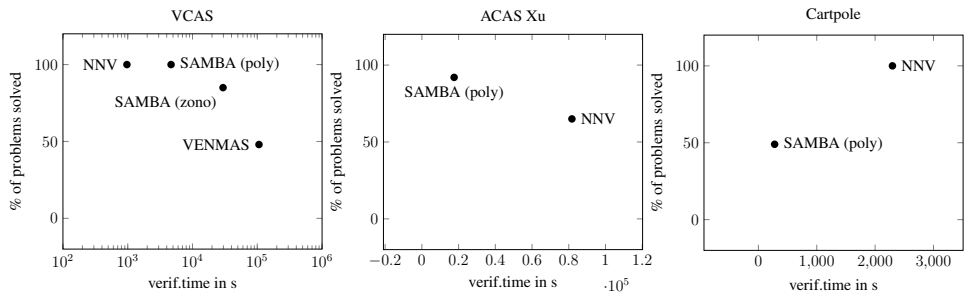


Figure 1: Comparaison des outils de vérification sur 3 cas d'étude: VCAS, ACAS Xu et Cartpole.

**Enseignements tirés** Le premier enseignement concerne l'applicabilité des méthodes formelles pour la vérification de CPSs basés machine learning : 97.3% des 225 problèmes de vérification considérés ont pu être résolus par l'un des outils. Un deuxième enseignement porte sur le coût de l'approche système étudiée dans cette article qui, si elle permet de s'affranchir de la problématique liée la spécification des réseaux de neurones, s'avère aussi beaucoup plus couteuse que l'analyse d'un réseau de neurones *isolé* (avec NNV, 95% du temps de résolution est consacré à l'analyse de la dynamique tandis que l'analyse des réseaux de neurones représente les 5% restants). Parmi les enseignements tirés, sont aussi discutées les possibles heuristiques utilisées dans SAMBA pour pallier sa moins bonne précision par rapport à NNV, ainsi qu'une heuristique pour le choix de la méthode la plus adaptée pour un problème de vérification donné.

# References


[1] Michael E. Akintunde, Elena Botoeva, Panagiotis Kouvaros, and Alessio Lomuscio. Verifying strategic abilities of neural-symbolic multi-agent systems. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR'20)*, pages 22–32, 2020.

[2] Julien Alexandre dit Sandretto and Alexandre Chapoutot. Validated Explicit and Implicit Runge-Kutta Methods. *Reliable Computing electronic edition*, 22, July 2016.

[3] Arthur Clavière, Eric Asselin, Christophe Garion, and Claire Pagetti. Safety verification of neural network controlled systems. In *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pages 47–54, 2021.

[4] Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin Vechev. AI $^2$: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 3–18, 2018.

[5] Kyle D. Julian, Mykel J. Kochenderfer, and Michael P. Owen. Deep neural network compression for aircraft collision avoidance systems. *Journal of Guidance, Control, and Dynamics*, 42(3):598–608, 2019.

[6] Diego Manzanas Lopez, Taylor Johnson, Hoang-Dung Tran, Stanley Bak, Xin Chen, and Kerianne L. Hobbs. Verification of neural network compression of acas xu lookup tables with star set reachability. In *AIAA Scitech 2021 Forum*.

[7] Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. *Proc. ACM Program. Lang.*, 3(POPL), 2019.

[8] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Formal security analysis of neural networks using symbolic intervals. In *27th USENIX Security Symposium, USENIX Security 2018*, pages 1599–1614, 2018.