Application du Machine Learning pour la détection d'anomalies dans les processus industriels

Florian Billon, Jonathan Brown et Jean-Christophe Testud

Sentryo, 66 Boulevard Niels Bohr, 69100 Villeurbanne
security@sentryo.net
https://sentryo.net

Résumé Nous proposons une méthode à base de Machine Learning pour détecter les attaques informatiques sur les processus industriels. L'impact physique de ces attaques pouvant être considérable, nous souhaitons que la détection se fasse au plus tôt. Pour ce faire, nous adoptons une technique de suivi de variables d'automates visibles sur le réseau. La différence entre les valeurs observées et celles prédites par l'algorithme développé dans le cadre de cette étude permet d'expliquer l'origine de l'anomalie. Cette approche permet une certaine interprétabilité, tout en étant basée sur un algorithme complexe. La méthodologie présentée ici a été conçue pour s'adapter à tous types de réseaux industriels et se déployer de manière autonome.

Keywords: Détection d'anomalies · Machine Learning · ICS · Sécurité · Intelligence artificielle · Processus industriels · Suivi de variables.

1 Introduction

1.1 Systèmes industriels

Les systèmes de contrôle industriels (ICS) deviennent de plus en plus complexes et connectés. Ils ont désormais la possibilité d'être reconfigurés rapidement et de remonter des informations sur l'état du processus industriel. Ces transformations impliquent souvent une connectivité forte entre les automates qui pilotent les composants de la chaîne de production et les stations informatiques qui vont programmer et suivre leurs actions. Cette connectivité se retrouve également entre les systèmes d'acquisition et de contrôle de données (SCADA) et le réseau externe. Cela augmente la surface et le nombre de vecteurs d'attaque. Une des conséquences de cette connectivité est la possibilité qu'un attaquant puisse s'introduire dans un réseau industriel et entreprendre des actions malveillantes. Par exemple, celui-ci peut altérer l'intégrité des valeurs industrielles remontées au SCADA, ce qui déclencherait une action incorrecte et potentiellement dangereuse. C'est notamment ce qui s'est passé avec Stuxnet [6], un ver informatique qui visait spécifiquement les systèmes SCADA pour reprogrammer des automates Siemens et manipuler les variables de pilotage des centrifugeuses

d'un centre d'enrichissement d'uranium iranien.
Le contexte industriel est particulier dans le sens où il regroupe de nombreux protocoles parfois propriétaires. Ces derniers permettent d'articuler en temps réel les automates programmables industriels (PLC), les SCADA et divers autres équipements. Nous présenterons ici une méthodologie pour détecter une tentative d'altération de processus industriel.

1.2 État de l'art

Il existe des systèmes de détection d'intrusion (IDS) dédiés aux ICS. Ceux-ci se basent sur des signatures, c'est-à-dire des comportements théoriquement empruntés par un attaquant (ouvertures de ports, contenus spécifiques de messages...). L'outil Snort [2] fonctionne sur ce principe. À l'inverse de cette approche se situent les systèmes de détection d'anomalie. Ceux-ci se basent sur la compréhension fine du comportement nominal du système. Dès lors que ce dernier s'écarte d'un comportement de référence, une alerte est émise.
Dans le contexte industriel, il est fréquent de retrouver des systèmes de détection d'anomalie configurés manuellement (à l'instar des IDS). Ces systèmes peuvent par exemple être implémentés à travers la définition de seuils au-delà desquels les valeurs sont considérées comme anormales. Ces valeurs sont récupérées via les flux de supervision qui requêtent les automates ("polling").
Récemment, différentes recherches ont étudié l'utilisation du Machine Learning pour la détection d'anomalies dans les systèmes industriels [7, 8, 10].

1.3 Emploi du Machine Learning

Aspects théoriques Le Machine Learning est un domaine à la croisée de l'informatique et des mathématiques dont le but est de donner la faculté à un système informatique d'apprendre à résoudre une tâche spécifique à partir de données, sans programmation explicite. Au travers d'un algorithme, une réponse (aussi appelée prédiction) peut être donnée à partir de données d'observation (représentées par une distribution de probabilité a priori inconnue).
Ce mapping $y = f(X)$ entre observations (X) et sortie de l'algorithme (y) est possible grâce à l'exploitation des corrélations entre les différentes variables observées. Cette phase s'appelle l'apprentissage du modèle. Les paramètres du modèle sont ajustés afin de répondre au mieux à la problématique posée, c'est-à-dire obtenir la meilleure prédiction possible pour la détection d'anomalies.
Dans un second temps, un algorithme entraîné est appliqué sur un nouveau jeu de données indépendant de celui utilisé lors de l'entraînement, il s'agit de la phase d'inférence. Il est possible de quantifier la performance du modèle au travers d'une métrique choisie de manière appropriée. Nous allons voir comment ces algorithmes peuvent être appliqués dans le domaine de la cybersécurité industrielle.

Cas d'usages et applications Les algorithmes peuvent être utilisés pour des cas d'usages différents :

— la classification, où la donnée d'entrée est divisée en deux ou plusieurs classes connues a priori. Cela est le cas par exemple d'un détecteur de spam où la donnée d'entrée représente des messages et la cible à prédire est la classe "spam" ou "non spam".
— la régression, où la valeur à prédire n'est pas une classe mais une valeur continue. Cela peut être fait pour prédire la valeur d'une variable d'automate à partir d'observations passées, comme dans le travail présenté ici.
— la détection de valeurs aberrantes pour détecter des cas rares où la donnée ne suit pas le comportement du reste de la population, comme celui d'un attaquant prenant des actions suspectes sur un poste de travail.

Les algorithmes de Machine Learning peuvent être entraînés de deux manières distinctes :

— Apprentissage supervisé : les algorithmes dits supervisés permettent de répondre aux problématiques de classification et de régression. Pour ce faire, les données d'observation ainsi que les réponses souhaitées (classe ou valeur à prédire) sont présentées à l'algorithme lors de l'apprentissage.
— Apprentissage non-supervisé : cet autre type d'apprentissage est employé dans des situations où l'on souhaite grouper des données entre elles, sans avoir connaissance de la classe à laquelle ils appartiennent.

1.4 Comparaison entre IDS classiques et Machine Learning

Depuis des décennies, des méthodes pour la sécurité informatique ont été développées en s'appuyant sur de simples règles métier provenant d'experts du domaine. Potentiellement, ces règles ne couvrent pas totalement la surface d'attaque ou peuvent initier un nombre élevé de faux positifs. Habituellement, un processus industriel peut contenir des dizaines de milliers de variables. La mise en place manuelle de seuils ne permet pas un passage à l'échelle pour un nombre trop élevé de valeurs. Le principe inhérent du Machine Learning permet de s'affranchir de ces limitations.
Ces algorithmes atteignent des performances élevées en reposant sur un principe différent de la programmation classique. Ils apprennent de manière implicite des règles métier à partir d'observations passées (données réseau collectées). Le Machine Learning, appliqué avec succès dans différents domaines d'application, est de plus en plus utilisé dans celui de la cybersécurité. Certaines limitations existent cependant.
Il y a naturellement moins de données représentant les attaques que les comportements normaux, il s'agit d'un problème pour lequel les classes à prédire sont fortement déséquilibrées. Ce manque de représentativité impacte les performances des approches usuelles de classification. De plus, les conséquences d'une erreur de décision sont critiques pour la cybersécurité. Pour pallier ces problèmes, nous utiliserons une méthode de détection d'anomalie basée sur l'apprentissage du comportement nominal du processus industriel.

2 Méthode générale pour le suivi de variables industrielles

Dans les processus industriels complexes, de nombreuses variables clés sont accessibles en lecture et en écriture entre les systèmes de supervision et les automates. Nous utilisons une méthode passive basée sur la surveillance du réseau (et l'analyse des paquets) pour obtenir les valeurs des variables au cours du temps. Il est donc possible, à partir des informations transitant sur le réseau, d'avoir une représentation partielle de l'état physique courant du processus industriel (seule une portion des variables manipulées par l'automate sont lues par la supervision).

2.1 Focus sur les variables d'automates

Les variables manipulées par les automates sont souvent liées à des processus physiques. Ces variables peuvent être fortement corrélées. Par exemple, lorsque la température dans une cuve augmente, la valeur de la pression augmente également. Les variables d'automates peuvent généralement être regroupées en trois catégories :

- les variables discrètes qui modélisent souvent l'état d'un actionneur,
- les variables continues associées à des capteurs,
- les variables constantes dans la fenêtre d'observation (souvent associées à des seuils).

La Figure 1 présente l'évolution de deux variables d'automates au cours du temps (les variables proviennent du jeu de données SWaT, présenté en 3.1). La variable `AIT203` (en bleu) correspond à un capteur de mesure du pH, c'est donc une variable continue. La variable `P101` (en orange) correspond à l'actionneur d'une pompe de transfert de l'eau, c'est donc une variable discrète. Ce graphique permet également de mettre en avant que les séries temporelles qui représentent les processus industriels sont non stationnaires. Ce comportement est représentatif du mode de fonctionnement d'une usine qui doit se réguler pour rester en état d'équilibre tout en respectant les contraintes métier (horaires, différentes productions, périodes de maintenance...).

2.2 Principe général

Le suivi de valeurs consiste à apprendre ces corrélations via l'observation des valeurs des variables. À partir d'une période d'observation, nous prédisons à l'aide d'un modèle de Machine Learning les valeurs des variables pour des pas de temps futurs. Le système surveille en temps réel la différence entre les valeurs prédites et celles observées sur le réseau afin de détecter des anomalies. Une anomalie se matérialise par une déviation significative par rapport à un signal nominal. Notre approche permet également de réduire la quantité de travail nécessaire pour définir les règles de détection tout en améliorant la capacité d'adaptation de l'outil. Le modèle a le potentiel de redécouvrir automatiquement les règles qu'un expert aurait configurées (comme un seuil de température), voire d'en découvrir de plus complexes (corrélation entre plusieurs variables).

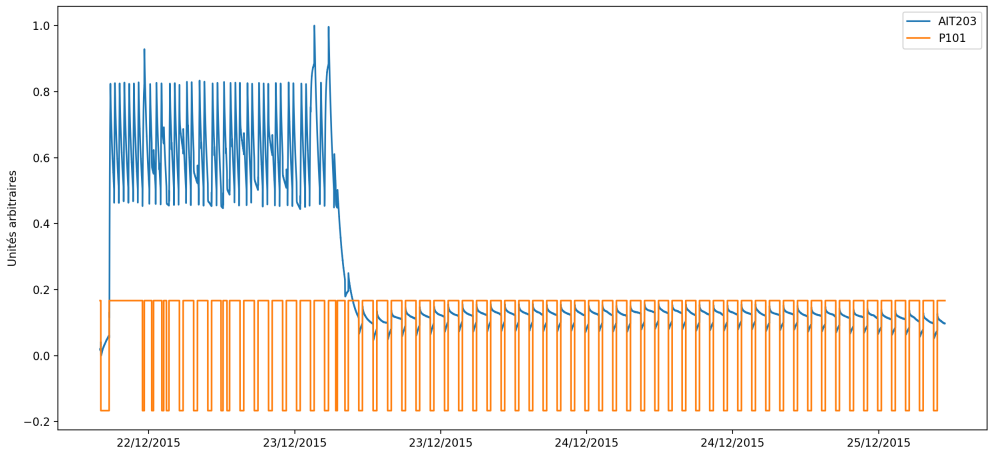

Figure 1. Exemple d'évolution des valeurs prises par une variable continue et une variable discrète. Les valeurs sont représentées sur une échelle arbitraire. La variable `AIT203` (en bleu) correspond à un capteur de mesure du pH. La variable `P101` (en orange) représente l'état d'un actionneur.

3 Implémentation sur un cas réel

Afin de développer une solution versatile qui puisse s'adapter de façon autonome à des systèmes industriels variés, la méthode est conçue en prenant en compte deux contraintes fortes :

— les données et les équipements sur lesquels le modèle doit s'appliquer sont inconnues,

— le déploiement, l'entraînement et l'application du modèle doivent se faire de manière autonome.

L'application bout-en-bout de la méthodologie sur des données réelles est présentée de manière détaillée dans cette section.

3.1 Présentation du jeu de données

Dans le cadre de notre étude nous avons utilisé un jeu de données correspondant à un système de traitement de l'eau (Secure Water Treatment ou SWaT) fourni par iTrust [11]. Il s'agit d'un système physique maquetté. Celui-ci est donc moins complexe qu'un système industriel classique. Ce jeu de données représente 11 jours de capture du réseau : 7 jours de données nominales et 4 jours de captures durant lesquels différents scénarios d'attaques ont été réalisés. Les parties nominale et attaque sont respectivement composées de 496802 et 449921 points de mesures. Les valeurs des 51 capteurs et actionneurs sont disponibles à raison

d'une valeur par seconde pendant toute la période.
Chaque pas de temps est horodaté et possède une classe (normal/attaque), ce qui permet de quantifier de manière assez précise les capacités de détection de notre approche. La diversité des attaques réalisées permet également d'avoir une idée plus précise de la capacité de détection pour chaque type d'attaque. Parmi ces attaques, nous pouvons par exemple observer des changements d'états d'actionneurs (un ou plusieurs en simultané) ou des valeurs de consignes anormales (augmentation continue du niveau de l'eau, dépassement d'un seuil de sécurité...).

3.2 Choix des variables

Les réseaux industriels peuvent compter plusieurs centaines d'automates, chacun pouvant posséder des centaines ou milliers de variables. Il n'est pas possible de surveiller l'ensemble de ces valeurs, aussi il est nécessaire de restreindre le périmètre du système de détection. Pour choisir une heuristique de sélection efficace des variables descriptives du problème, nous avons fait l'hypothèse qu'il fallait, dans un premier temps, sélectionner un unique automate. Pour sélectionner les variables les plus pertinentes, nous nous basons principalement sur le type des variables observées (discrètes, continues ou constantes).
Les variables constantes possèdent peu d'intérêt et polluent le modèle en ajoutant une complexité non nécessaire à l'algorithme. Les variables discrètes associées aux actionneurs peuvent paraître nécessaire pour mieux comprendre les relations entre variables, cependant dans de nombreux cas les changements d'états sont trop rapides pour être tous lus par la supervision et donc ne peuvent être capturés. Les données sont donc incomplètes et le modèle pourrait apprendre des corrélations erronées. Il est donc important de sélectionner en particulier les variables continues parmi l'ensemble des variables proposées. Pour cela nous utilisons une heuristique simple, le nombre de valeurs distinctes pris par la variable, comme proposé dans [9]. Si ce nombre de valeurs est supérieur ou égal à 8, nous considérons une variable comme continue et nous la sélectionnons pour modéliser le processus industriel. À titre indicatif, cette heuristique a identifié 25 variables continues, ce qui correspond exactement aux capteurs de l'installation.

3.3 Algorithme

L'algorithme utilisé est un réseau de neurones. Le principe général de ce type d'algorithme est de pouvoir exploiter les corrélations entre variables afin de faire une prédiction. Dans le cas plus précis présenté ici, un type particulier de réseau de neurones est employé, il s'agit d'un réseau de neurones récurrents. Sa particularité est de pouvoir traiter des données temporelles [3]. Cela est possible car, à la différence des réseaux de neurones classiques, les Gated Recurrent Unit (GRU) [4,5] employés dans ce cas se servent d'une mémoire interne afin de traiter et trouver des motifs séquentiels à partir de données ordonnées dans le temps. La librairie `keras` [1] implémente ce type d'algorithmes.
Le réseau choisi est défini par N_c couches intermédiaires, chacune composée

de N_n neurones. De plus, la taille des séquences temporelles prises en entrée est définie par N_t pas de temps. L'algorithme prédit en sortie des séquences de N_t pas de temps. Le choix de ces paramètres de l'algorithme est présenté maintenant.

3.4 Choix des paramètres

L'architecture du réseau de neurones a été choisie de manière à fournir de bons résultats tout en gardant un temps d'entraînement raisonnable. Pour cela, une seule couche composée de 512 neurones a été choisie ($N_c = 1$ et $N_n = 512$). La taille des séquences prises en entrée de l'algorithme est aussi un paramètre qui peut être ajusté. Nous avons donc réalisé une étude où cette valeur a été variée afin de minimiser le taux de faux positifs (nombre de séquences incorrectement labellisées comme attaque par rapport au nombre total de séquences). Le résultat est montré sur la figure 2. Il apparaît qu'une taille de séquence de $N_t = 100$ est optimale. Cette exercice a été répété avec la métrique de précision ($\frac{vp}{vp+fp}$, où vp et fp correspondent respectivement aux nombres de vrais et faux positifs) et mëme résultat a été obtenu.

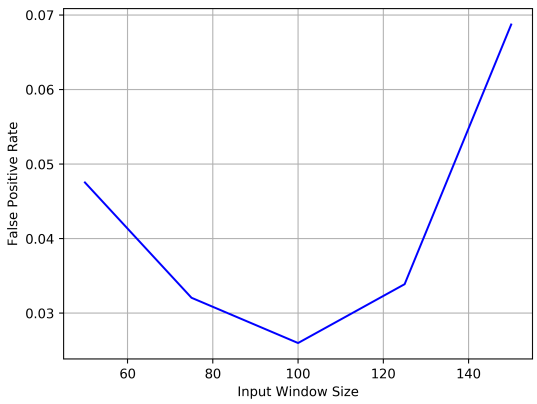

Figure 2. Évolution du taux de faux positifs en fonction de la taille de la fenêtre.

Un autre paramètre peut être choisi sur l'échantillonnage de la donnée. La période entre chaque pas de temps est de 1 seconde. Afin d'éviter d'être en présence de données trop bruitées et pour réduire la taille du jeu de données, les variables industrielles sont agrégées temporellement en prenant la valeur moyenne pour chacune d'entre elles dans une fenêtre de 3 secondes. Cela signifie que pour $N_t = 100$, nous prédisons un intervalle de temps de 5 minutes.

3.5 Choix d'un seuil de détection

Comme expliqué précédemment, le principe de la détection d'anomalie consiste à comparer une valeur observée à partir du réseau industriel avec une valeur prédite par l'algorithme présenté plus haut. Si la différence Δ entre ces deux valeurs est significativement différente, l'utilisateur devra être averti. Δ est défini comme l'erreur quadratique entre deux séquences. Plus précisément, l'erreur δ_t pour un pas de temps t fixé est :

$$\delta_t = \frac{\sum_{v \in V} (\widehat{Y}_{t,v} - Y_{t,v})^2}{|V|}$$

où V correspond à l'ensemble des variables et $\widehat{Y}_{t,v}$ et $Y_{t,v}$ aux valeurs respectivement mesurées et prédites de la variable v au temps t. La valeur Δ est alors calculée comme :

$$\Delta = \frac{\sum_t \delta_t}{N_t}$$

Une valeur seuil sur Δ doit être établie pour signifier à l'utilisateur la présence d'une anomalie. Nous constatons que Δ fluctue fortement d'un pas de temps à un autre. Afin de stabiliser cette valeur au cours du temps, un algorithme de lissage est employé pour s'affranchir de cet effet indésirable qui pourrait ainsi causer un nombre de faux positifs élevés. Le lissage est effectué en prenant la valeur médiane de Δ prise pour 100 séquences consécutives. La figure 3 illustre l'effet de ce lissage sur l'erreur brute.

La valeur maximale Δ_{max} obtenue sur un jeu de données de validation est alors choisie comme seuil d'alerte. Lors de la phase de déploiement, une anomalie sera signalée lorsque la valeur Δ d'une séquence (après lissage) est supérieure au seuil établi.

3.6 Résultats

En suivant les étapes présentées dans cet article, un algorithme est entraïné puis appliqué au jeu de données comportant les attaques (nous nous sommes concentrés sur la première journée du jeu de données d'attaque). La valeur du seuil estimé à partir du jeu de données de test est de 1.606. En appliquant cette valeur seuil sur les données d'attaque, le taux de faux positifs est de 0.011, la métrique Précision vaut quant à elle 0.612.

La Figure 4 montre l'évolution de la variable industrielle `LIT101`. On constate que le modèle parvient efficacement à reproduire le comportement de la variable en dehors des périodes d'attaque. A l'inverse, les prédictions faites durant les attaques ne suivent pas les valeurs lues sur le réseau, impliquant que l'erreur Δ dépasse la valeur seuil δ_{max}. Une période d'anomalie est ainsi détectée.

La majorité des attaques se situent au début de la période analysée. La modélisation de la variable est impactée durant cette phase, alors que progressivement les prédictions redeviennent conformes au comportement attendu, à mesure que le processus industriel se stabilise après les attaques. Aussi, bien que certaines

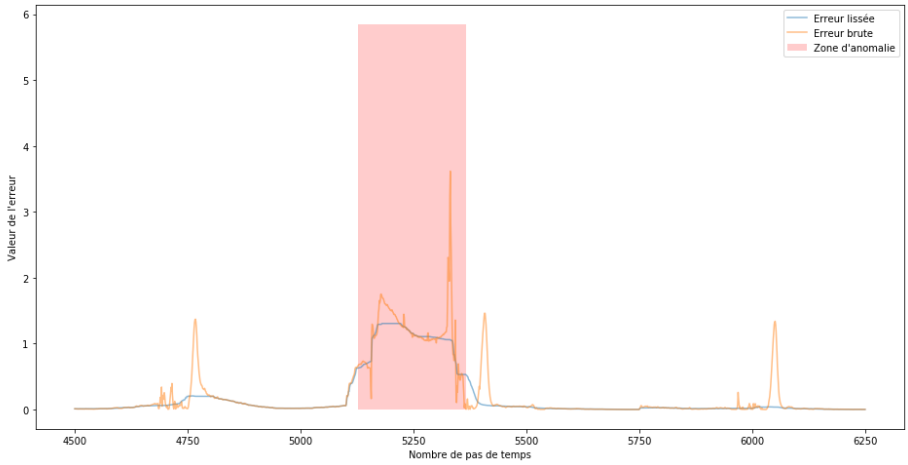

Figure 3. Illustration de la différence entre l'erreur brute et l'erreur lissée par une médiane glissante.

périodes d'attaques soient bien identifiées comme des anomalies, l'algorithme ne parvient pas à toutes les détecter. On constate aussi plusieurs heures après les phases d'attaque la présence de 3 fausses alertes de courte durée.

4 Contraintes de déploiement et processus de mise en service

Les résultats présentés précédemment concernent des données pour lesquelles nous avions la possibilité d'effectuer l'ensemble des opérations manuellement et localement. Après chaque étape principale (sélection de variables, entraînement du modèle, application de celui-ci) nous avons pu analyser les résultats et modifier, si nécessaire, différents paramètres. Pour nous adapter à des situations plus contraignantes et automatiser notre démarche, nous avons mis en place un processus de mise en service complet qui ne nécessite pas d'intervention après son lancement (Figure 5). Notons que l'ensemble des données en entrée sont des données nominales. Plusieurs étapes de ce processus sont identiques aux points présentés précédemment, notamment les phases de sélection de variables ou de définition du seuil de détection. Voici quelques informations concernant les autres points.

Récolte de données Les données sont généralement récoltées directement sur le réseau, par exemple à l'aide de sondes de captures placées au niveau de différents switchs. Le fonctionnement classique de requêtage des variables par la supervision implique que les valeurs des variables ne sont pas nécessairement récupérées à chaque pas de temps. Aussi, une phase de préparation de la donnée

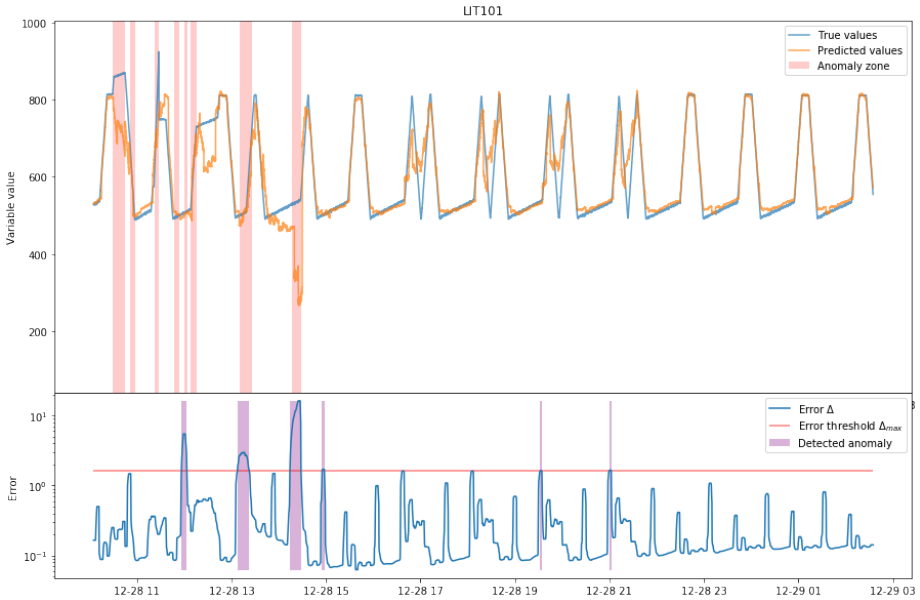

Figure 4. Illustration de la variable industrielle LIT101. En haut : valeurs lues sur le réseau et prédites par le modèle (respectivement en bleu et jaune). Les zones d'attaques sont représentées en rouge. En bas : sur une échelle logarithmique, l'erreur de la prédiction Δ est montrée en bleu et le seuil δ_{max} en rouge. Les périodes où une anomalie est détectée est en violet.

permet d'uniformiser les séries temporelles représentant les variables industrielles sélectionnées. Dans des cas réels, les données manquantes sont complétées en prenant les valeurs précédentes ou via une interpolation.

Homogénéité de l'erreur de prédiction Nous vérifions que l'erreur sur les données de test est comparable à celle sur les données d'entraînement. Pour ce faire, nous définissons une limite supérieure de 25% de différence. Cela permet de valider que les données utilisées pour l'entraînement sont suffisamment représentatives du processus industriel à modéliser et que le modèle se généralise bien à des données nouvelles. Cette heuristique reste à affiner.

Comparaison à un modèle de référence Afin de vérifier que le modèle lève des alertes pertinentes, sa performance est comparée par rapport à un modèle "naïf" servant de référence. Ce modèle prédit le même comportement que celui observé en entrée (fonction de délai). Cette étape permet de vérifier que l'algorithme identifie des corrélations entre les variables et dans le temps.

Détection d'une attaque injectée Afin de vérifier que notre modèle est capable de détecter des anomalies, nous générons différents scénarios : remplace-

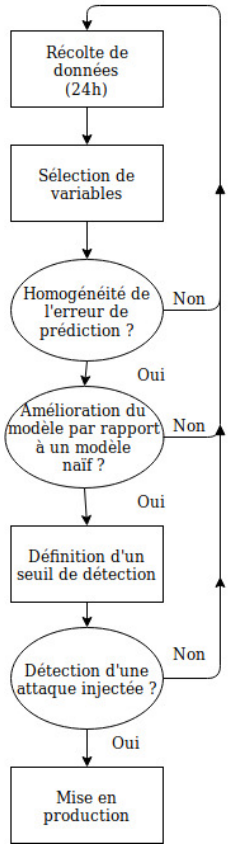

Figure 5. Schéma du processus de mise en service.

ment d'une variable par une valeur constante, dépassement de la valeur maximale observée, remplacement d'une variable par des valeurs aléatoires dans l'intervalle des valeurs possibles. Ces scénarios peuvent impacter une ou plusieurs variables.

5 Perspectives

5.1 Interprétabilité

L'une des principales critiques à l'usage d'algorithmes de Machine Learning est le manque de compréhension et d'interprétabilité du modèle. Ceci est particulièrement problématique lors de l'application à des domaines critiques (comme la sécurité des systèmes industriels). L'avantage des alertes générées par notre approche par rapport à des alertes binaires est de proposer à l'utilisateur une visualisation des variables les plus impactées. Pour ces variables, l'opérateur pourra observer leur comportement attendu et l'écart avec le comportement mesuré.

5.2 Améliorations

Le taux de faux positifs obtenu dans les résultats sur SWaT est encourageant mais doit encore être amélioré pour une utilisation en milieu réel. Pour répondre à ce problème, plusieurs axes d'amélioration sont envisageables. D'une part, d'autres algorithmes de Machine Learning méritent également une étude approfondie. Par exemple, l'utilisation d'algorithmes bayésiens profonds permet de donner des intervalles de confiance sur la présence d'une anomalie. De plus, une intervention des experts guiderait l'algorithme dans son apprentissage afin de réduire le nombre de faux-positifs tout en améliorant le choix des données d'entrée en utilisant des compétences "métier". D'autre part, les paramètres utilisés doivent faire l'objet d'une attention particulière afin de s'assurer qu'ils sont optimaux. Pour ce faire, nous pouvons utiliser une phase automatisée d'optimisation basée sur des données d'entraînement hétérogènes. Cependant, les captures réseau relatives aux milieux industriels restent très peu répandues, aussi il est difficile de s'assurer de la capacité de généralisation de notre modèle.

6 Conclusion

Dans ce papier, nous avons présenté une approche pour la détection d'attaques dans des réseaux industriels basée sur des algorithmes innovants de Machine Learning. Celle-ci se fait par la comparaison des variables industrielles observées sur le réseau et les valeurs prédites par un réseau de neurones récurrents de type GRU. Le taux de faux positifs obtenu est de 0.011, et la métrique Précision est de 0.612.
Une méthodologie a aussi été développé pour déployer automatiquement l'algorithme de détection d'anomalie sur un environnement inconnu. Une séquence d'étapes à satisfaire a été détaillée afin de s'assurer de la robustesse de l'algorithme. Cette méthode a aussi l'avantage de présenter des résultats interprétable, au-delà d'un score binaire peu explicable.

Références

1. Keras. `https://keras.io/`. Accessed : 2018-09-28.
2. Snort. `https://www.snort.org`. Accessed : 2018-09-28.
3. Understanding LSTM networks. `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`. Accessed : 2018-09-28.
4. Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.
5. Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.
6. Nicolas Falliere, Murchu, and Eric Chien. W32.Stuxnet Dossier. Symantec Security Response online report, February 2011.

7. Pavel Filonov, Fedor Kitashov, and Andrey Lavrentyev. Rnn-based early cyber-attack detection for the tennessee eastman process. *CoRR*, abs/1709.02232, 2017.
8. Jonathan Goh, Sridhar Adepu, Marcus Tan, and Zi Shan Lee. Anomaly detection in cyber physical systems using recurrent neural networks. *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*, pages 140–145, 2017.
9. Dina Hadžiosmanović, Robin Sommer, Emmanuele Zambon, and Pieter H. Hartel. Through the eye of the PLC : Semantic security monitoring for industrial processes. In *Proceedings of the 30th Annual Computer Security Applications Conference*, ACSAC '14, pages 126–135, New York, NY, USA, 2014. ACM.
10. Jun Inoue, Yoriyuki Yamagata, Yuqi Chen, Christopher M. Poskitt, and Jun Sun. Anomaly detection for a water treatment system using unsupervised machine learning. *CoRR*, abs/1709.05342, 2017.
11. iTrust, Centre for Research in Cyber Security, Singapore University of Technology and Design. Secure water treatment (SWaT). `https://itrust.sutd.edu.sg/testbeds/secure-water-treatment-swat/`, note = Accessed : 2018-09-28.