Nouvelle approche de diagnostic en ligne des Systèmes Automatisés de Productions

Ramla Saddem, Dylan Baptiste, Achraf Marrakh

Nouvelle approche de diagnostic en ligne des Systèmes Automatisés de Productions

Application au système Import-Export de la CellFlex du CReSTIC

Ramla Saddem, Dylan Baptiste et Achraf Marrakh
CReSTIC UFR Sciences Exactes et Naturelles Reims, France

ramla.saddem@univ-reims.fr

Résumé

Dans ce papier nous nous intéressons au diagnostic en ligne des Systèmes Automatisés de Production possédant des capteurs et des actionneurs délivrant des signaux binaires Tout ou Rien (TOR) et qui relèvent des Systèmes à Evénements Discrets. Nous proposons une solution de diagnostic intelligente afin de remplacer les solutions traditionnelles souvent non industrialisables par une nouvelle méthode à base de données apprises depuis la simulation de comportements normaux et/ou anormaux depuis un jumeau numérique, en utilisant les réseaux de neurones récurrents (RNN) à mémoire court-terme et long terme (Long short-term memory, LSTM).

Mots clés : Diagnostic**,** industrie du futur, Système Automatisé de Production, Machine Learning, LSTM, RNN

1 Introduction

Le contexte de ce travail est le diagnostic des Systèmes Automatisés de Production (SAP). Dans un contexte « industrie du futur », les systèmes de production se doivent d'être plus flexibles et résilients tout en devenant plus complexes. Les exigences de performance (production, qualité, sécurité) conduisent les industries à anticiper les différents phénomènes de défaillance. Dans les activités de maintenance, on cherche alors à mettre en œuvre des solutions dites de « Prognostics and health Management » (PHM). Selon la norme NF EN 13306 (2010), la maintenance est définie comme un « ensemble de toutes les actions techniques, administratives et de management durant le cycle de vie d'un bien, destinées à le maintenir ou à le rétablir dans un état dans lequel il peut accomplir la fonction requise ». Elle comprend ainsi un ensemble d'actions de surveillance, dépannage/réparation, de vérification pour l'amélioration des processus. Afin d'assurer la sûreté de fonctionnement des biens et matériels, la tâche de diagnostic consiste à détecter, isoler, identifier au plus juste et au plus tôt la moindre défaillance ou écart du comportement nominal d'une machine.

Les systèmes auxquels nous nous intéressons dans cet article sont les Systèmes Automatisés de Production (SAP) de type Systèmes à événements Discrets (SED). Les approches de diagnostic des SED classiques de la littérature reposent principalement sur :

1) des études hors ligne de la diagnosticabilité d'un système (capacité à diagnostiquer un défaut avec certitude dans un temps fini),
2) des modèles observateurs du système en ligne à intégrer au sein du processus de commande.

Ces approches « diagnostiqueurs », bien connues de la communauté nécessitent cependant un travail d'expertise important afin d'obtenir des modèles du système performant et s'exposent rapidement à un problème d'explosion de l'espace d'états à observer et rend par ailleurs le calcul des diagnostiqueurs à implémenter en ligne souvent impossible pour les systèmes complexes (Lafortune, 2019).

Nous souhaitons opposer cette approche dite à base de modèles à une proposition basée sur l'apprentissage automatique de données du système, et ce que ce soit pour un comportement normal comme anormal. Pour cela, nous proposons d'utiliser un des outils de l'industrie du futur : le jumeau numérique. La notion de jumeau numérique (Kritzinger et al., 2018) (Tao et al., 2019) consiste à digitaliser une usine et à reproduire numériquement son comportement. La plupart des solutions industrielles permet de faire correspondre un comportement désiré de la machine pour en faire une mise en service virtuelle (Virtual Commissioning). Ici, nous souhaitons l'utiliser afin d'injecter des défaillances dans le système numérisé pour enrichir son apprentissage.

L'objectif de ce travail est de présenter une approche pour le diagnostic en ligne des SAP qui ont une dynamique discrète. Notre solution se base sur des méthodes issues du domaine de l'intelligence artificielle (IA) et utilise des techniques d'apprentissage automatique pour diagnostiquer l'occurrence de fautes dans un SAP en ligne.

2 Bref aperçu de l'état de l'art

Dans ce contexte, nous nous intéressons au diagnostic des comportements indésirables dans les SAP. La littérature propose différentes approches traitant de cette problématique. Ces approches se répartissent selon la dynamique des SAP en trois classes : la classe des systèmes continus (Cellier and Kofman, 2006), la classe des systèmes à événements discrets (SED) (Cassandras and Lafortune, 2009) et la classe des systèmes dynamiques hybrides (SDH) (Zaytoon et al., 2001). Dans cet article, nous nous intéressons au diagnostic des SAP possédant des capteurs et des actionneurs délivrant des signaux binaires Tout ou Rien (TOR) et qui relèvent des SED. Les approches de diagnostic pour cette catégorie de systèmes peuvent être vues selon que le diagnostic s'effectue en ligne ou non (Sampath et al., 1995), (Boussif and Ghazel, 2021), selon que le modèle soit spécifié (par automate ou par réseau de Petri) ou non (Zaytoon and Lafortune, 2013), (Basile, 2014), selon la structure de prise de décision du diagnostic (centralisée, décentralisée ou distribuée), selon la représentation des fautes et leurs reconnaissances (Zaytoon and Lafortune, 2013), etc. En général, les approches se classent en trois grandes familles selon le mode du raisonnement utilisé pour le diagnostic : les approches à base de modèles (Sampath et al., 1995), (Debouk et al., 2000), (Zaytoon and Lafortune, 2013), les approches à base de connaissances (Dousson et al., 1993), (Bertrand, 2009), (Saddem et al., 2012) et les approches à base de données (Venkatasubramanian et al., 2003), (Moosavian et al., 2013), (Dou and Zhou, 2016), (Vazan et al., 2017), (Han et al., 2017).

Les approches à base de modèles sont généralement utilisées dans le cadre d'une connaissance suffisante du fonctionnement interne du système. Elles sont efficaces et capables de valider la cohérence et la complétude des défauts à diagnostiquer. Cependant, pour fonctionner correctement, ces approches nécessitent des modèles analytiques précis et approfondis du domaine et la difficulté majeure est le coût élevé de mise en œuvre des modèles (Danancher et al., 2011), (Saddem and Philippot, 2014) (De Souza

et al, 2020). En effet, la complexité temporelle de mise en œuvre de la plupart des modèles est une complexité exponentielle.

Les approches à base de connaissances ont une capacité de diagnostic élevée à cause de la connaissance des symptômes des fautes qu'elles modélisent (Cordier and Dousson, 2000). Néanmoins, la difficulté majeure réside dans la formalisation des connaissances expertes et dans leurs mises à jour (Guerraz and Dousson, 2004), (Dousson et al., 2008), (Cram et al., 2012), (Subias et al., 2014), (Saddem and Philippot, 2014). Ainsi, pour que les approches remplissent leurs missions convenablement, elles nécessitent une base de connaissance complète et cohérente.

Les approches à base de données (Venkatasubramanian et al., 2003), (Moosavian et al., 2013), (Dou and Zhou, 2016), (Vazan et al., 2017), (Han et al., 2017) ne nécessitent pas de connaitre de manière approfondie le fonctionnement interne du système. Elles n'ont pas besoin d'une modélisation explicite du système à travers un modèle mathématique. Elles utilisent les données d'historiques disponibles et à partir de ces données, elles font des prévisions. Ces approches apprennent de chaque expérience afin d'améliorer leurs performances. Elles s'appuient sur les techniques de l'apprentissage automatique pour atteindre ses objectifs. Toutefois, elles nécessitent une étape de préparation de données afin d'extraire les données les plus pertinentes qui vont être formatées selon la technique d'apprentissage automatique à utiliser. Dans ce papier nous proposons une approche de diagnostic à base de donnée qui à notre connaissance n'a pas été proposée auparavant.

3 Approche proposée

3.1 Système automatisé de Production

Un SAP se compose de deux parties : la partie opérative (PO) et la partie commande (PC). La partie opérative représente l'ensemble des moyens matériels opérant physiquement sur la matière d'œuvre.

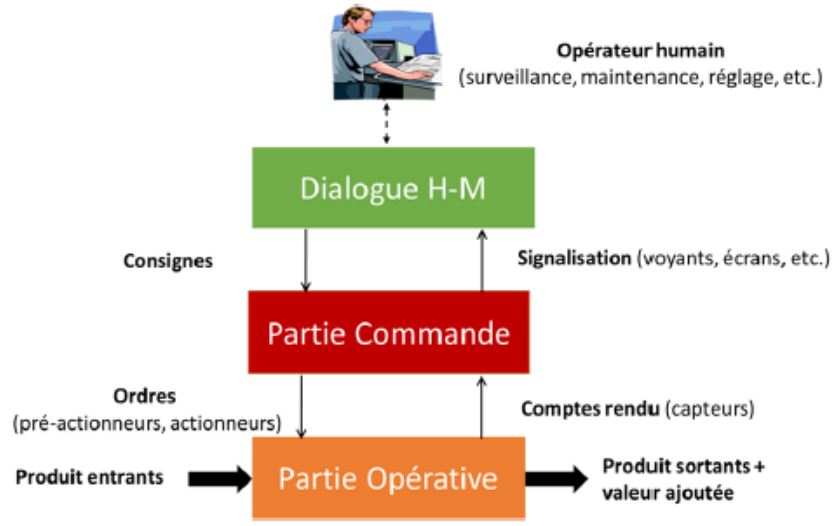


Figure 1 : Structure d'un SAP

La partie commande est l'ensemble des moyens de traitement et d'acquisition de l'information qui assurent le pilotage et la conduite du procédé. Les échanges d'information entre la PC et la PO sont de deux types : i) La PC envoie les ordres vers les actionneurs et les pré-actionneurs de la PO afin d'obtenir

les effets souhaités ii) La PO envoie les comptes rendus vers la PC à travers les capteurs. La partie dialogue Homme Machine (H-M) qui est présentée dans la Figure 1 permet la communication entre la PC et les opérateurs humains. En effet, ces opérateurs échangent les informations avec la PC à travers une interface. Ils donnent des consignes à l'aide d'IHM. En parallèle, la PC renvoie des signalisations diverses telles que les indicateurs lumineux, les indicateurs sonores, les messages affichés sur les écrans, etc.

La plupart des SAP qui possèdent des capteurs et des actionneurs délivrant des signaux binaires (Tout ou Rien), et qui se classent dans les SED, sont commandés par des automates programmables industriels (API) qui effectuent trois étapes de base :

a) l'acquisition des entrées qui consiste à la réalisation d'une image du monde extérieur à travers l'enregistrement des états des entrées qui sont des variables non contrôlables (capteurs).

b) l'exécution du programme de la commande.

c) la mise à jour des états des sorties qui sont des variables contrôlables (actionneurs).

Ces étapes s'effectuent cycliquement c'est-à-dire enchainant les cycles les uns après les autres. Le diagnostic consiste par conséquent à acquérir cycliquement les informations issues des capteurs et des ordres de commande, les transformer en événements observables et les analyser afin de détecter et d'isoler les fautes à partir de l'occurrence des évènements observables.

3.2 Proposition

Dans cet article, nous proposons une solution originale pour le diagnostic en ligne des SAP qui ont une dynamique discrète. Notre solution se base sur des méthodes issues du domaine de l'intelligence artificielle (IA). Ainsi, nous utilisons des techniques d'apprentissage automatique (Machine Learning (ML)) pour diagnostiquer l'occurrence de fautes dans un SAP en ligne ou de son jumeau numérique. Le développement et le déploiement des modèles d'apprentissage automatique impliquent une série d'étapes (Figure 2):

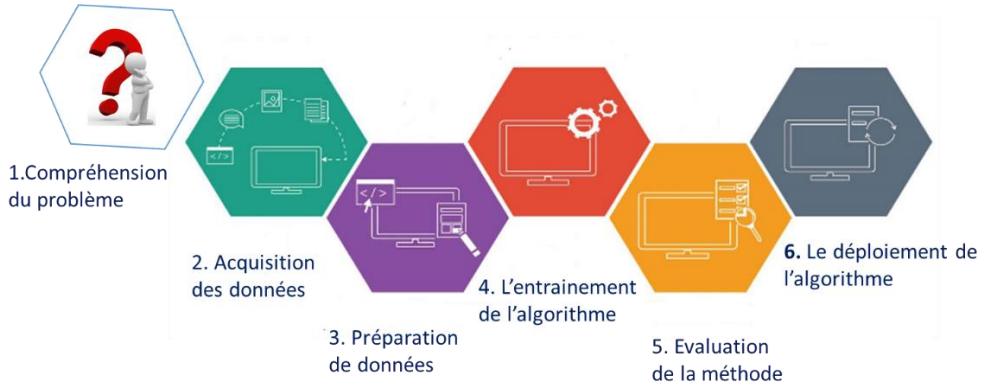


Figure 2 : Etapes de l'apprentissage automatique

i. La définition du problème consiste à comprendre le problème à résoudre, à déterminer les objectifs (prédiction, regroupement, etc.), à définir les critères du succès et les contraintes à respecter.

ii. L'acquisition de données consiste à identifier et collecter les données nécessaires pour supporter le problème. Ces données peuvent provenir de plusieurs sources et peuvent être

structurées (telles que les enregistrements des bases de données, les arbres, les graphes, etc) ou non structurées (telles que les images, les textes, les voix, etc).

iii. La préparation des données qui consiste en la mise en forme des données selon l'algorithme d'apprentissage automatique à utiliser. Elle inclut la transformation, la normalisation, le nettoyage et la sélection des données d'apprentissage.

iv. L'entraînement de l'algorithme d'apprentissage avec les données d'apprentissage et de validation qui consiste à diviser les données disponibles en trois groupes (données d'apprentissage, données de validation et données de test). L'algorithme est entraîné avec les données d'apprentissage. Ensuite, les données de validation sont utilisées pour régler les hyper paramètres qui sont des paramètres dont la valeur est définie avant le début de la phase d'apprentissage. Les données de test sont utilisées pour les tests.

v. Le test de l'algorithme sur les données de test qui consiste à vérifier les performances de l'algorithme sur les données de test. Si ces performances sont bonnes, nous pouvons passer à l'étape suivante.

vi. Le déploiement de l'algorithme.

3.3 Compréhension du problème

Pour cette étape, une étude et une analyse du système est nécessaire : définition de la liste des composants du SAP et un cahier de charges de fonctionnement du SAP. Dans ce travail nous nous intéressons au diagnostic en ligne des SAP possédant des composants : capteurs et actionneurs délivrant des signaux binaires TOR. Quatre types de fautes peuvent survenir sur un composant : un collage à 1 ; un collage à 0 ; un passage de 0 à 1 inattendu et un passage de 1 à 0 inattendu. Le SAP surveillé peut être normal, défaillant ou incertain. L'état incertain signifie que le système peut être normal ou défaillant : on ne dispose pas d'assez d'indicateurs pour décider s'il est normal ou défaillant. L'objectif donc est de retourner l'état de santé du SAP surveillé en ligne. Si une faute d'un composant du SAP survient le diagnostic retourne cette faute. Pour cela, nous devons disposer de la liste des composants du SAP surveillé pour fixer le nombre et le nom de chaque faute qui peut avoir lieu. Un cahier de charges de fonctionnement du SAP permet d'établir une commande de fonctionnement normal. Nous supposons que la commande ne contient pas de fautes, càd que si la PC du SAP envoie l'ordre à un actionneur de passer à une valeur binaire, nous supposons que cet ordre arrive correctement à la PO.

3.4 Acquisition des données

Pour cette étape, nous avons utilisé l'application AIDMAP II qui permet l'acquisition et l'enregistrement des valeurs de composants du SAP surveillé. AIDMAP II s'appuie sur la norme OPC, acronyme de " Ole for Process Control ". Cette norme détermine un standard de communication universel et performant entre les équipements automatisés et les applications d'informatique industrielle. Les enregistrements sont des fichiers auxquels on peut accéder à partir de n'importe quel ordinateur connecté au réseau. Les serveurs OPC dialoguent avec AIDMAP II par le même canal, qu'il s'agisse d'un réseau local ou d'un réseau étendu. Le résultat de cette étape est un fichier Excel où sont enregistrés les changements de valeurs des variables surveillées. Chaque ligne représente le nom de la variable, la date d'occurrence de changement et sa valeur vrai ou faux.

Nous commençons par collecter les données du SAP en mode normal. Ensuite nous injectons dans le jumeau numérique une faute à la fois et nous collectons les données du jumeau numérique. Plusieurs scénarios sont possibles : nous commençons la simulation puis nous injectons la faute ou alors nous injectons la faute puis nous commençons la simulation. L'opération est répétée jusqu'à obtention d'un nombre suffisant de données. Ce nombre est arbitraire dans ce travail. Nous avons fait pour chaque

scénario 10 simulations différentes. Le nombre de simulations possible est infini donc nous nous sommes contentés de 10 simulations de chaque faute injectée. Il est important de noter que les fautes qu'on a simulées avec le jumeau numérique sont des collages à 0 ou à 1 des composants. Les fautes représentant les passages inattendus d'une valeur à une autre, font partie des extensions possibles de ce travail.

3.5 Préparation des données

Dans cette étape, nous transformons les fichiers Excel obtenus à l'étape 2 en un fichier contenant des lignes de la forme présentée dans la Figure 3. Ensuite, selon l'architecture utilisée dans l'étape de l'entraînement, selon le nombre d'étapes passées à donner à notre réseau de neurones, nous formatons les données en tant que valeur étiquetée. Par exemple, nous choisissons de donner en entrée à l'algorithme d'apprentissage les 50 états passés et on souhaite prédire le 51ème. Les valeurs True deviennent des 1 et les valeurs false deviennent 0. La date de chaque état est modifiée pour être relative au dernier état et les données sont mises à l'échelle entre 0 et 1.

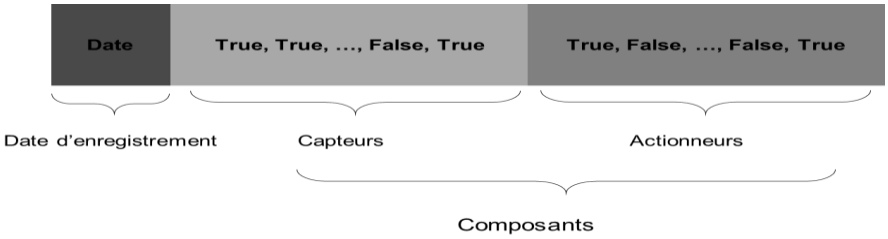


Figure 3 : représentation d'un état

3.6 L'entraînement de l'algorithme d'apprentissage

Pour entraîner l'algorithme d'apprentissage, nous avons utilisé les réseaux de neurones récurrents (RNN) à mémoire court-terme et long terme (Long short-term memory, LSTM) (Hochreiter, 1997), pour prédire les valeurs des composants du SAP à l'instant t+1 et comparer le résultat avec le vecteur de sortie quand il a lieu. Les RNN sont un type de réseau de neurones particulièrement efficace pour la gestion de données temporelles ou tout type de séquences de données. Ils sont très utilisés dans le domaine du NLP (Natural Language Processing) que ce soit pour la traduction, la classification de texte, le résumé de documents, la génération de texte etc. Ils restent cependant typologiquement similaires à un réseau de perceptrons.

Dans ce travail, nous utilisons des couches de neurones LSTM adaptés à la gestion séquentielle de nos données. LSTM est un modèle neuronal efficace pour un grand nombre d'applications impliquant des données temporelles ou séquentielles (Karpathy, 2015) comme par exemple l'analyse vidéo (Donahue, 2015), la reconnaissance de la parole (Graves et al, 2013), la modélisation du langage (Mikolov, 2010), la reconnaissance de l'écriture manuscrite ou sa génération (Graves, 2013), la traduction automatique (Bahdanau, 2014), (Sutskever, 2014), les sous-titrage des images (Vinyals, 2015), (Karpathy, 2017) etc.

La Figure 4 représente l'architecture du modèle RNN. Les valeurs du vecteur prédit sont des valeurs comprises entre 0 et 1 (Figure 5). Ainsi en arrondissant la partie du composant, nous avons la prédiction (Figure 6). La couche de sortie sera activée avec la fonction sigmoïde.

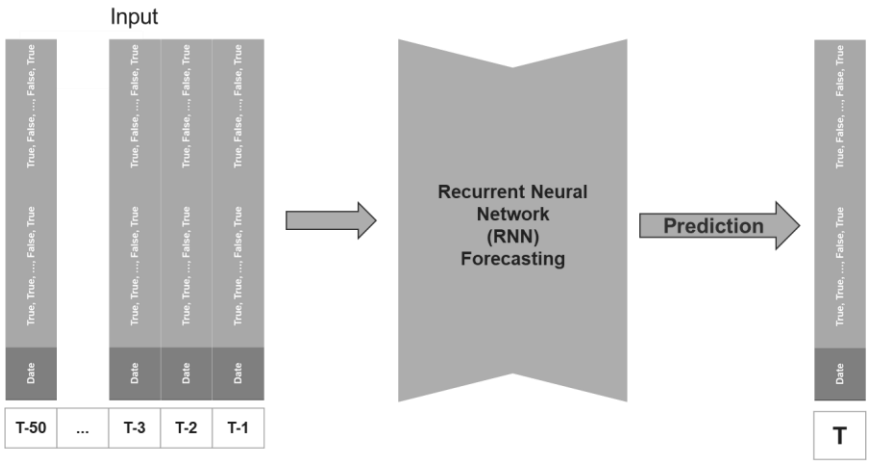


Figure 4 : Architecture des entrées sorties du modèle RNN

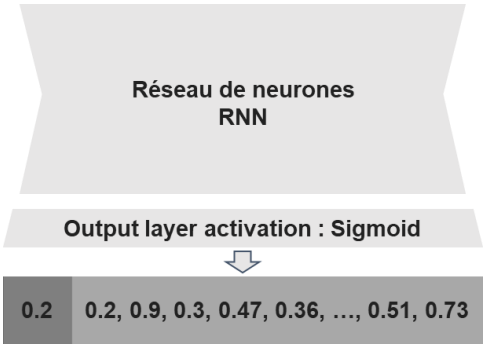


Figure 5 : Format du vecteur prédit avant arrondissement

Figure 6 : Format du vecteur prédit après arrondissement

4 Application sur le système réel

4.1 Description du système

Le système sur lequel nous appliquons notre approche est le système Import-Export de la CellFlex. Avant de donner la description de ce système (paragraphe 4.1.2), nous présentons l'ensemble de la CellFlex dans le paragraphe suivant pour donner une idée sur le fonctionnement général du système global.

4.1.1 La CellFlex

La CellFlex est un élément principal de la plate-forme de formation et de recherche CELLFLEX4.0 de l'Université de Reims Champagne-Ardenne. La cellule flexible, appelée CellFlex, est un groupe de huit stations fonctionnant ensemble autour d'un convoyeur central. Ces stations simulent le fonctionnement d'une ligne de production de mise en bouteille, sous la forme d'une usine miniaturisée reliée sur un réseau composé de standards industriels. Une représentation schématique de la cellule est présentée sur la figure 7.

4.1.2 La station Import-Export

Dans ce travail, nous nous sommes intéressés à la station Import-Export, numéro 6 sur la figure 7.

a) Mode de fonctionnement

- Import : consiste à insérer, sur le convoyeur central, un nouveau 6-pack vide. Lorsqu'un 6-pack est présent en entrée du convoyeur d'importation, il doit être acheminé à la fin du convoyeur, en dessous du vérin vertical. Lorsqu'une palette vide se présente à la bonne position sur le convoyeur centrale (capteur palette zone 6), le 6-pack en attente est chargé sur la palette.

- Export : L'exportation consiste à prendre depuis le convoyeur central un 6-pack plein. Lorsqu'un 6-pack est présent sur le convoyeur central, face à la station, les vérins doivent être mis en mouvement afin de prendre le 6-pack, puis le déposer sur une des deux lignes d'exportation. L'exportation est prioritaire sur l'importation, ceci afin d'éviter un blocage du système. Il existe deux lignes d'exportation, celles-ci sont inclinées permettant d'acheminer les 6-packs facilement vers la sortie de la station. La ligne 1 doit être remplie en priorité pour optimiser le temps d'exportation. Lorsque la ligne 1 est pleine, la ligne 2 doit être utilisée pour l'exportation.

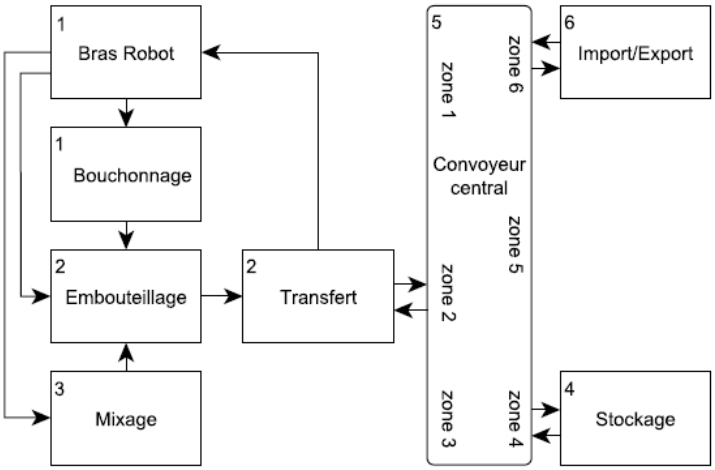
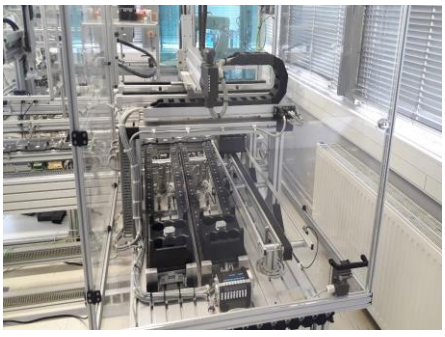
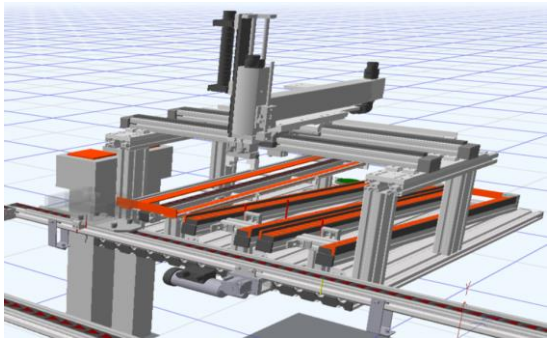


Figure 7 : Les huit stations de la CellFlex

b) Description



a) Station Import-Export (Système réel)



b) Station Import-Export (Système Virtuel)

Figure 8 : Description de la station d'import/export

La station Import-Export (Figure8) consiste en : un vérin double-effet pour chaque axe de déplacement (X, Y, Z), un convoyeur permettant l'arrivée de nouveaux 6-packs, deux lignes inclinées permettant l'exportation de 6-packs, des petits vérins assurant la stabilité durant la pose d'un 6-pack sur les lignes inclinées, un petit vérin bloqueur central permettant de positionner le système au-dessus de la ligne d'exportation 1, et d'une pince (accrochée au vérin vertical) permettant d'agripper les 6-packs. Le vérin vertical est équipé d'un frein. Celui-ci empêchant tout déplacement lorsqu'il n'est pas relâché. On voit sur La Figure 8 la station Import-Export (à gauche le système réel et à droite son jumeau numérique, le système virtuel).

4.2 Résultats

Dans ce paragraphe, nous présentons les résultats de l'application de notre approche sur le système import-Export de la CellFlex. Pour l'étape compréhension du problème (paragraphe 3.3), nous avons développé le programme de commande de la station Import-Export. Nous avons utilisé TIA PORTAL V15.1. Le jumeau numérique que nous avons utilisé a été développé avec Emulate 3D. Pour l'étape acquisition des données (paragraphe 3.4), nous avons lancé le système en mode normal. Nous avons collecté avec AIDMAP II les changements de valeurs de variables pendant plusieurs cycles. Notons que le nombre de cycle varie d'un SAP à un autre. Pour notre système, le cycle dure en moyenne quelques minutes. Nous avons jugé que l'enregistrement des changements de valeurs de variables pendant quelques heures est suffisant. Ensuite, nous avons injecté une faute à la fois dans le jumeau numérique. En effet, Emulate 3D nous permet de forcer un capteur ou un actionneur à une valeur (vrai ou faux). Ce forçage simule un collage à 1 ou à 0 du capteur ou de l'actionneur en question. Nous avons collecté avec AIDMAP II les changements de valeurs de variables pendant plusieurs cycles après l'injection de la faute. Notre système est composé de 23 capteurs et 10 actionneurs. Nous avons simulé le collage à 1 et le collage à 0 de chaque capteur et de chaque actionneur. Pour la partie préparation des données, nous avons transformé les fichiers Excel obtenus à l'étape précédente en un fichier contenant des lignes de la forme présentée dans la Figure 3. Ensuite, nous avons formaté les données en tant que valeur étiquetée comme décrit dans le paragraphe 3.5. Le vecteur de la Figure 3 est composé de la date et de 33 valeurs booléennes. Pour l'entraînement de l'algorithme d'apprentissage, nous avons choisi arbitrairement d'utiliser 50 valeurs passées pour prédire la 51ème valeur du vecteur d'entrée sortie (Figure 4). La couche de sortie de RNN sera activée avec la fonction sigmoid. Pour la fonction d'erreur, en ce qui concerne la partie des composants qui est de type multi-label, nous utilisons la fonction Binary Cross-Entropy (BCE). En ce qui concerne la partie du temps qui est un problème de régression classique, nous utilisons la fonction Mean Square Error (MSE).

Pour entraîner le modèle, nous avons utilisé la fonction d'erreur suivante :

$$e(P) = (P_t - R_t)^2 * K_1 + \left(-\frac{1}{N}\sum_{i=1}^{N} R_{c_i} * \log\left(P_{c_i}\right) + \left(1 - R_{c_i}\right) * \log\left(1 - P_{c_i}\right)\right) * K_2$$

Avec K1 et K2 $\in$ [0,1] les coefficients d'ajustement

P_t, R_t : temps Prédit et temps Réel

Pc, Rc : vecteur Prédit et vecteur réel

N : Nombre de composants

La Figure 9 représente le modèle utilisé (une couche d'entrée, trois couches cachées avec dix nœuds chacune et une couche de sortie). La couche d'entrée respecte la forme des données : 50 états passés de 34 valeurs (33 composants (23 capteurs et 10 actionneurs) et le temps. Les trois couches cachées sont composées des cellules LSTM. La couche de sortie respecte la forme de la prédiction (le vecteur de l'état futur). Le choix du nombre de nœuds et du nombre de couches cachées font partie des nombreux hyperparamètres. Le choix a été fait arbitrairement.

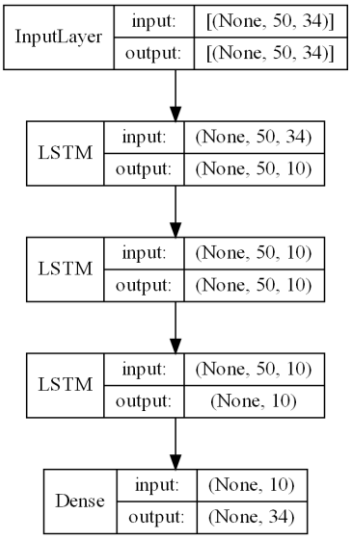


Figure 9 : Modèle1 du RNN

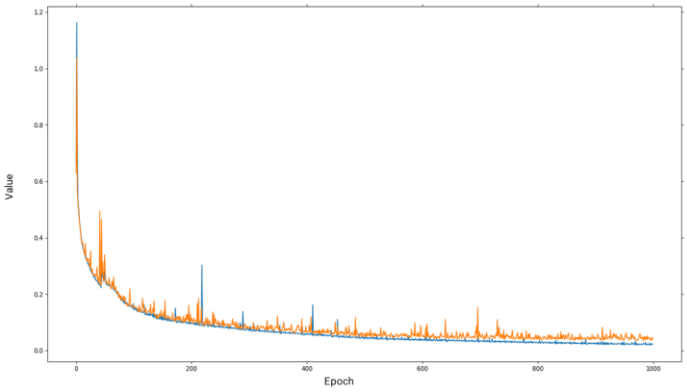


Figure 10 : Evolution de la fonction d'erreur du modèle1

Nous avons choisi d'ignorer l'erreur de prédiction du temps car ce qui nous intéresse est la valeur de vecteurs d'entrées/sorties. Nous avons donc annulé la valeur de K1 dans la fonction d'erreur (K1=0) et nous avons choisi K2=1. La Figure 10 illustre les résultats obtenus : l'évolution de la fonction d'erreur qui converge vers zéro. Nous avons mesuré l'accuracy de notre model. L'accuracy est la mesure du taux de la bonne classification. Pour cela nous avons utilisé deux mesures d'accuracy différentes composant-acc1 et composant-acc2 (Tableau 1). Dans le tableau 1 nous avons illustré les deux mesures d'accuracy sur un système simple de trois composants pour expliquer ces mesures. En bleu, les bonnes valeurs prédites, en rouge les mauvaises valeurs prédites. Composant_acc1 mesure le nombre de composants correctement prédits indépendamment les uns des autres. Et Composant_acc2 mesure le nombre d'états correctement prédits.

Mesure	Terrain => Prediction
	[0, 1, 0] => [0, 1, 0] **[0, 1, 1] => [1, 1, 0]** **[1, 1, 0] => [0, 0, 0]**
Composant_acc1	**5 composants / 9** **55%**
Composant_acc2	**1 ligne / 3** **33%**

Tableau 1 : Mesures d'accuracy

Nous pouvons constater que les résultats sont bons. L'accuracy2 est évidemment plus difficile à être proche de 100% mais il semble qu'avec plus d'époques d'entraînement cela soit possible. Comme l'historique d'entraînement du modèle n'a pas montré de signe de surapprentissage et qu'une meilleure convergence vers 0 semblait possible, nous avons multiplié par dix le nombre de nœuds de chaque couche cachée (toujours un choix arbitraire). Ce qui a donné le nouveau modèle de RNN présenté dans la Figure 12. Le nombre de neurones dans les couches cachées est égal à 100. Chaque époque d'entraînement devrait prendre plus de temps mais moins d'époques pour avoir de meilleurs résultats.

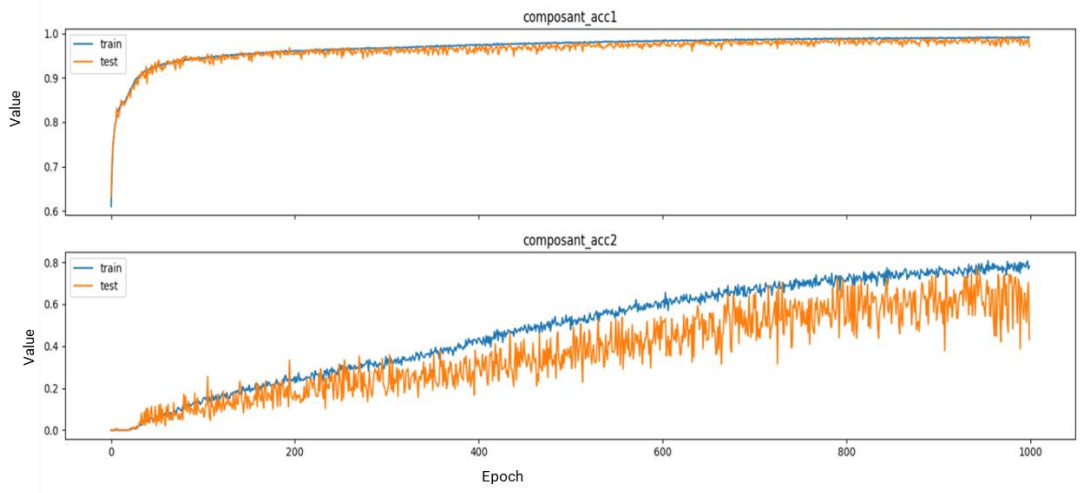


Figure 11 : Evolution de l'accuracy du modèle de la Figure 9 par rapport aux époques de l'entraînement

La Figure 14 montre de meilleurs résultats. L'accuracy1 est de 99,9% en entraînement et en test. L'accuracy2 est de 99% pour l'entraînement et 92% pour le test. Quant à l'évolution de la valeur de la fonction d'erreur, on peut remarquer que la courbe associée aux données de test se sépare de celle des données d'entraînement vers la 400ème époque et que ce phénomène semble s'accentuer. Cela traduit un phénomène bien connu de surapprentissage (overfitting): le modèle se spécialise sur les données d'entraînement et ne parvient plus à généraliser ses prédictions sur des données sur lesquelles il ne s'est jamais entraîné. Il existe de multiples manières de minimiser ce phénomène (Dropout par exemple) qui sera une perspective de ce travail.

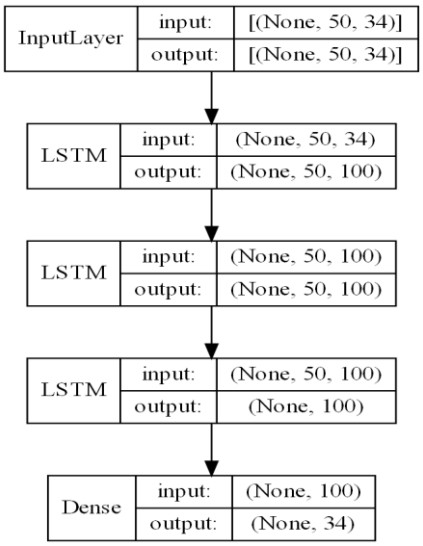


Figure 12 : Modèle2 du RNN

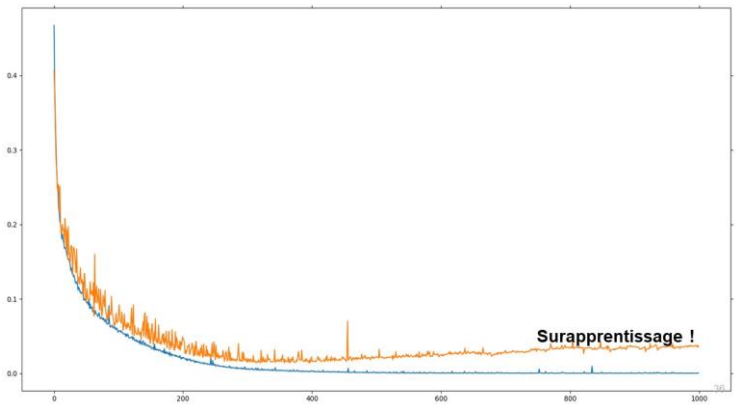


Figure 13 : Evolution de la fonction d'erreur pour Binary Cross-Entropy du modèle2

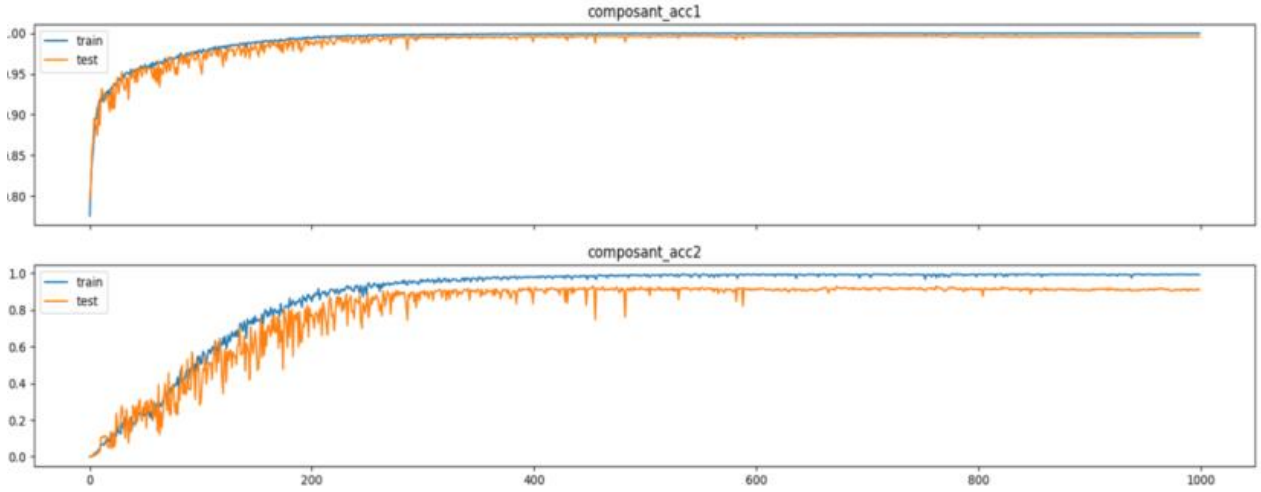


Figure14: Evolution de l'accuracy du modèle de la Figure 12 par rapport aux époques de l'entraînement

Plus le modèle est complexe (le nombre de paramètres entraînables est important), mieux il est capable de bien modéliser un problème donné mais il risque la surinterprétation/surapprentissage. Aussi plus le modèle est complexe, plus les époques d'entraînement sont longues à calculer. Un modèle qui n'est pas assez complexe pour un problème donné risque le sous-apprentissage (mauvaise convergence vers 0 de la fonction d'erreur).

5 Conclusion

Dans ce papier nous avons proposé une nouvelle approche pour le diagnostic en ligne des SAP de la classe SED. Cette nouvelle approche est à base de données. L'acquisition de données des comportements normaux et défaillants se fait depuis un jumeau numérique en utilisant l'outil AIDMAP II. Grâce au projet de recherche de plateformes interconnectées Factories of Future Champagne-Ardenne (FFCA), financé dans le cadre du CPER 2018-2020, le CReSTIC dispose aujourd'hui de plusieurs Jumeaux Numériques de l'atelier flexible de cellflex4.0. La préparation des données consiste en la transformation des fichiers issus de AIDMAP II en vecteurs pour le modèle RNN proposé. Les résultats de l'application de la méthode proposée sur le système Import-export de la CellFlex montrent l'apport et l'intérêt de celle-ci.

Plusieurs perspectives sont envisageables. Dans ce travail, le nombre de couches cachées et le nombre de neurones dans les couches cachées du RNN sont choisis arbitrairement. Un travail de comparaison de différentes valeurs est envisageable. Le nombre d'états passés est aussi choisi arbitrairement, 50. On pourrait le réduire ou l'augmenter. Soit n ce nombre. Nous pouvons aussi prédire les m prochains vecteurs en connaissant les n états passés. Il faudra alors adapter la préparation des données, adapter la fonction d'erreur (qui ressemblera à une double moyenne par partie (temps, composants)) et adapter les mesures d'accuracy. Enfin, nous avons appliqué l'approche sur le système Import-export de la cellFlex qui contient sept autres stations, une application sur les autres stations est envisageable.

Références

Bahdanau, Cho, D. K. Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. CoRR, vol. abs/1409.0473.

Basile, F. (2014). Overview of fault diagnosis methods based on petri net models. Proc. of European Control Conference (ECC), IEEE, 2636-2642.

Bertrand, O. (2009). Détection d'activités par un système de reconnaissance de chroniques et application au cas des simulations distribuées HLA. PhD thesis, Paris 13.

Boussif, A. and Ghazel, M. (2021). Tuning the Diagnoser-based Approach for Diagnosability Analysis of Finite Automata. International Journal of Control, Automation and Systems, 19 (8), 2842-2858.

Cassandras, C. G. and Lafortune, S. (2009). Introduction to discrete event systems. Springer Science & Business Media.

Cellier, F. E. and Kofman, E. (2006). Continuous system simulation. Springer Science &Business Media.

Cordier, M.-O. and Dousson, C. (2000). Alarm driven monitoring based on chronicles. IFAC Proceedings Volumes, 33(11) :291–296.

Cram, D., Mathern, B., and Mille, A. (2012). A complete chronicle discovery approach : application to activity analysis. Expert Systems, 29(4) :321–346.

Danancher, M., Roth, M., Lesage, J.-J., and Litz, L. (2011). Diagnostic des SED basé sur un modèle : trois approches évaluées sur une même étude de cas. In 4èmes Journées Doctorales/Journées Nationales MACS (JD-JN-MACS'11), 165–170.

De Souza, R.P.C., Moreira, M.V. and Lesage, J.J. (2020). A hierarchical approach for discrete-event model identification incorporating expert knowledge. 15th Workshop on Discrete Event Systems, 53(4), 275-281.

Debouk, R., Lafortune, S., and Teneketzis, D. (2000). Coordinated decentralized protocols for failure diagnosis of discrete event systems. Discrete Event Dynamic Systems, 10(1-2) :33–86.

Dou, D. and Zhou, S. (2016). Comparison of four direct classification methods for intelligent fault diagnosis of rotating machinery. Applied Soft Computing, 46 : 459–468.

Donahue, J., Hendricks, L., Guadarrama, S. Rohrbach, M., Venugopalan, S., Darrell, T., Saenko. K. (2015, juin). Long-term recurrent convolutional networks for visual recognition and description. 2625–2634.

Dousson, C., Gaborit, P., and Ghallab, M. (1993). Situation recognition : representation and algorithms. In IJCAI, volume 93, 166–172.

Dousson, C., Clerot, F., and Fessant, F. (2008). Method for the machine learning of frequent chronicles in an alarm log for the monitoring of dynamic systems. US Patent 7, 388-482.

Graves, A. (2013). Generating Sequences With Recurrent Neural Networks. ArXiv, vol. abs/1308.0850.

Graves, A., Mohamed, A.R., Hinton, G. (2013, mars). Speech Recognition with Deep Recurrent Neural Networks. ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, vol. 38.

Guerraz, B. and Dousson, C. (2004). Chronicles construction starting from the fault model of the system to diagnose. In International Workshop on Principles of Diagnosis (DX04), 51–56. Citeseer.

Han, T., Jiang, D., Zhao, Q., Wang, L., and Yin, K. (2017). Comparison of random forest, artificial neural networks and support vector machine for intelligent diagnosis of rotating machinery. Transactions of the Institute of Measurement and Control, 2682-2693.

Hochreiter, H. and Schmidhuber, J. (1997, novembre). Long Short-Term Memory. Neural Comput., vol. 9, no. 8, 1735–1780.

Karpathy, A., Johnson J., Fei-Fei, L. (2015, juin). Visualizing and Understanding Recurrent Networks. Cornell Univ. Lab.

Karpathy, A., Fei-Fei, L., (2017, Avril). Deep Visual-Semantic Alignments for Generating Image Descriptions. IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 4, 664–676.

Kritzinger, W., Karner, M., Traar, G., Henjes, J., & Sihn, W. (2018). Digital Twin in manufacturing: A categorical literature review and classification. 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018, 51(11), 1016–1022.

Lafortune, S. (2019). Discrete event systems: Modeling, observation, and control. Annual Review of Control, Robotics, and Autonomous Systems, vol. 2, 141-159.

Moosavian, A., Ahmadi, H., Tabatabaeefar, A., and Khazaee, M. (2013). Comparison of two classifiers ; k-nearest neighbor and artificial neural network, for fault diagnosis on a main engine journal-bearing. Shock and Vibration, 20(2), 263–272.

Saddem, R., Armand, T., and Moncef, T. (2012). Algorithme d'interprétation d'une base de signatures temporelles causales pour le diagnostic en ligne des systèmes à événements discrets. In 9th International Conference on Modeling, Optimization & SIMulation.

Saddem, R. and Philippot, A. (2014). Causal temporal signature from diagnoser model for online diagnosis of discrete event systems. In Control, Decision and Information Technologies (CoDIT), 2014 International Conference on, 551–556. IEEE.

Sampath, M., Sengupta, R., Lafortune, S., Sinnamohideen, K., and Teneketzis, D. (1995). Diagnosability of discrete-event systems. IEEE Transactions on automatic control, 40(9) :1555–1575.

Subias, A., Travé-Massuyès, L., and Le Corronc, E. (2014). Learning chronicles signing multiple scenario instances. IFAC Proceedings Volumes, 47(3), 10397–10402.

Sutskever, I., Vinyals, O., Le, Q.V. (2014). Sequence to Sequence Learning with Neural Networks. In NIPS.

Tao, F., Qi, Q., Wang, L., & Nee, A. Y. C. (2019). Digital Twins and Cyber–Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison. Engineering, 5(4), 653–661.

Vazan, P., Janikova, D., Tanuska, P., Kebisek, M., and Cervenanska, Z. (2017). Using data mining methods for manufacturing process control. IFAC-PapersOnLine, 50(1) :6178–6183.

Venkatasubramanian, V., Rengaswamy, R., Kavuri, S. N., and Yin, K. (2003). A review of process fault detection and diagnosis : Part iii : Process history based methods. Computers & chemical engineering, 27(3), 327–346.

Vinyals, O., Toshev, A., Bengio S., Erhan, D. (2015, juin). Show and tell : A neural image caption generator. 3156–3164.

Zaytoon, J. et al. (2001). Systèmes dynamiques hybrides. Hermes.

Zaytoon, J. and Lafortune, S. (2013). Overview of fault diagnosis methods for discrete event systems. Annual Reviews in Control, 37(2), 308–320.