Détection d'anomalies dans des images de tunnels par forêts d'arbres aléatoires ou par apprentissage profond

G. DECOR[1,2] M. BAH[1,3] P. FOUCHER[1] P. CHARBONNIER[1] F. HEITZ[2]

[1] Équipe-projet ENDSUM, Cerema, Strasbourg
[2] ICube, UMR 7357, Université de Strasbourg, CNRS
[3] Centre d'Études des TUnnels (CETU), Bron

guillaume.decor@cerema.fr

Résumé

La maintenance des tunnels requiert une inspection visuelle lourde et contraignante. Dans le but d'automatiser cette tâche, nous proposons d'évaluer et de comparer trois algorithmes d'apprentissage statistique, une forêt aléatoire et deux réseaux convolutifs, dédiés à la détection d'anomalies (e.g. des fissures) sur les parements des tunnels. Chaque modèle est entraîné sur deux banques de données créées par nos soins, à partir d'images de parois en béton et de parois en maçonnerie. On montre qu'ils permettent tout trois d'atteindre le niveau de l'état de l'art sur ce domaine applicatif.

Mots Clef

Détection d'anomalies, Inspection de tunnels, Classification supervisée, Apprentissage profond, Forêts aléatoires.

Abstract

Tunnel maintenance requires a complicated and restrictive visual inspection. In order to automate this task, we propose to evaluate and compare three statistical learning algorithms, one random forest and two convolutional networks, dedicated to the detection of anomalies (e.g. cracks) on tunnel linings. Each model is trained on two databases of our own, from images of concrete walls and masonry walls. We show that they are competitive with the state of the art on this application domain.

Keywords

Anomaly detection, Tunnel inspection, Supervised classification, Deep learning, Random forests.

1 Introduction

Maintenir les tunnels (navigables, routiers et ferroviaires) en bon état d'usage est impératif pour garantir la sécurité des biens et des personnes qui les empruntent. Cela requiert de cartographier périodiquement les anomalies de surface (par exemple des fissures) pouvant entraîner des dégradations plus importantes (e.g. éclatement du béton mettant à nu les ferraillages), voire de mettre en péril la structure. Les phénomènes de dégradation des parements de tunnels sont variés, mais souvent liés à la présence non souhaitée d'eau, qu'il est donc également souhaitable de répertorier. Les figure 1 et 3 montrent quelques exemples d'anomalies (fissures, coulures, pertes de matière) sur des parements de tunnels en béton et en maçonnerie.

FIGURE 1 – Exemples d'images du piédroit d'un tunnel en maçonnerie (à gauche) et de la clé de voûte d'un tunnel en béton (à droite)

À l'heure actuelle, l'inspection des tunnels est effectuée par des agents spécialisés, directement sur site. Elle est contraignante, puisqu'elle requiert un temps d'intervention sur place important et un temps d'exploitation au bureau tout aussi conséquent. Dans ce contexte, le Cerema cherche à faciliter l'inspection visuelle des tunnels à travers la mise au point d'un dispositif d'acquisition d'images à haut rendement complété par un algorithme de traitement des images recueillies. D'un point de vue opérationnel, cela doit permettre de réduire le temps d'immobilisation des infrastructures ainsi que le temps consacré par les agents à l'auscultation des ouvrages d'art.

Détecter automatiquement des anomalies sur des images de parements de tunnels constitue une tâche particulièrement complexe, du fait de la variabilité des textures, formes, couleurs et objets d'intérêt à détecter. Cette variabilité se retrouve au sein de chacune des classes et il existe un recouvrement entre la classe des revêtements sains et la classe des défauts. En effet, suivant le type de matériau utilisé (béton, grès, craie, etc.) et la façon dont a été construit le revêtement (voussoir, béton projeté, etc.), l'aspect visuel de

la paroi va grandement différer. En fonction de ces paramètres, les défauts ne vont pas non plus se manifester de la même manière. Ainsi, une coulure sur un voussoir n'aura pas le même aspect que sur une construction en maçonnerie, car les sédiments déplacés ne sont pas les mêmes. Il en va de même pour les fissures, qui auront tendance a « rejoindre » les joints des moellons pour la maçonnerie tandis qu'elles présenteront une trajectoire arbitraire sur un voussoir. Les méthodes d'apprentissage statistique sont de bonnes candidates pour résoudre ce type de problèmes. Dans cet article, nous proposons d'évaluer, à partir de bases de données réelles constituées d'images acquises et annotées par nos soins, les performances de deux méthodes : les forêts aléatoires et les réseaux convolutionnels, pour la détection de défauts sur des images de parements en béton ou maçonnés (voir figure 1).
L'article est organisé de la manière suivante : après un tour d'horizon des différents algorithmes de reconnaissance des formes proposés à des fins d'inspection de tunnels (section 2), nous présentons dans la partie 3, les données expérimentales et les algorithmes que nous avons mis en œuvre. Les résultats de nos expérimentations sont ensuite détaillés et discutés dans la section 4. Enfin, nous proposons une conclusion et des perspectives (section 5).

2 Travaux associés

On peut observer qu'un nombre relativement limité de travaux a été consacré à la détection d'anomalies dans les tunnels par analyse d'images.

2.1 Algorithmes sans apprentissage

Les méthodes sans apprentissage que l'on trouve dans la littérature sont presque entièrement consacrées à la détection des fissures (voir par exemple [14]). Elles s'inspirent souvent de travaux sur des applications connexes comme celui de la détection de fissures sur chaussées. Elles font parfois appel à des méthodes relativement élémentaires (filtrages et seuillages des niveaux de gris, combinaisons d'opérateurs morphologiques, techniques par subdivisions successives et ajustement du point médian). Des méthodes plus élaborées se fondent sur les modèles déformables et la propagation de chemins minimaux (voir un état de l'art de ces techniques dans [1]), ou encore sur la méthode dite de *Percolation* [16] qui consiste à examiner la géométrie de la surface de niveau propagée depuis un germe.
Ces méthodes sont néanmoins restrictives, dans le sens où elles ne sont pensées que pour détecter des fissures sur des revêtements en béton ne présentant pas d'autres types de défauts. Les adapter pour augmenter la typologie des anomalies à détecter ou encore la diversité des types de parements supportés demanderait un effort conséquent.

2.2 Algorithmes avec apprentissage

Plusieurs algorithmes avec apprentissage ont été expérimentés ces dernières années. Les trois algorithmes présentés ci-après ont chacun été entraînés sur des banques de données différentes.
Dans le domaine de la détection de fissures, les auteurs de [12] suggèrent l'utilisation d'une architecture convolutive dans laquelle les images sont préalablement enrichies de descripteurs additionnels, non convolutifs. Ainsi, à partir d'une image en niveaux de gris $w \times h \times 1$, on construit une image $w \times h \times 6$ par l'ajout de caractéristiques telles que les histogrammes de gradients orientés (HOG) [4], des cartes de contours, ou d'entropie locale. Une exactitude de 88,6 % est obtenue sur une base de test de 10000 images (90000 images étant utilisées pour l'apprentissage et la validation).
Dans [3], le modèle, convolutif également, est construit en deux temps : on apprend un premier réseau convolutif. Après quoi, on remplace le perceptron multicouches par un classifieur SVM, que l'on entraîne avec les caractéristiques extraites des couches de convolution, dont les poids sont alors fixés. On parvient à une exactitude de 85,94 % en validation et 74,9 % sur le jeu de test. La base est constituée de 3000 images pour l'apprentissage et la validation et de 2800 pour les tests. La précision et le rappel s'élèvent respectivement à 82 % et 78% en test (contre 84 % et 79 % en validation).
Certains travaux cherchent à construire un modèle capable de détecter simultanément plusieurs types de désordres (fissures, venue d'eau et dépôt de matière). Une telle approche est décrite dans [5]. Une variante de ResNet [9] est employée. Cette dernière permet d'atteindre une exactitude de 87,5 % sur une base de 603 images dont 20 % sont dédiées au test.
Il ressort de ces articles une bonne performance de l'apprentissage (et, en particulier, des réseaux convolutionnels) quant à la détection d'anomalies dans les ouvrages d'art. On peut d'ailleurs noter la variabilité des données dans les bases d'apprentissage d'un modèle à l'autre, ce qui accrédite plus encore la thèse que les méthodes d'apprentissage sont adaptées à notre problématique métier.

2.3 Notre contribution

Dans cet article, nous mettons en œuvre deux réseaux convolutifs pour la détection d'anomalies sur une banque de données que nous avons nous-même constituée sur la base d'acquisitions réalisées par nos soins. Nous comparons les résultats à ceux obtenus préalablement à partir d'une classification avec des forêts aléatoires [6] sur les mêmes données. Nous montrons que ces modèles sont compétitifs au regard de l'état de l'art dans le domaine de la détection d'anomalies sur les parements des tunnels.

3 Méthodologie

3.1 Données

Pour construire des jeux d'apprentissage, le Cerema a réalisé ses propres acquisitions sur le terrain. Des parements de tunnels, routiers et navigables, ont été photographiés au moyen d'un système d'acquisition (montré sur la figure 2) composé de 6 caméras et embarqué sur un véhicule. Chaque image est en 1080p (i.e. 1920×1080 pixels)

FIGURE 2 – Système de prises de vues du Cerema monté sur une automobile (à gauche) ou un bateau (à droite) pour l'acquisition d'images en tunnels routiers ou navigables

		Apprentissage	Test	Total
Béton	Sain	594	77	671
	Défaut	600	201	801
	Total	1194	278	**1472**
Maçonnerie	Sain	508	302	810
	Défaut	508	268	776
	Total	1016	570	**1586**

TABLE 1 – Composition (nombre d'exemples) des deux bases d'imagettes

et couvre environ 5 m^2 de paroi (voûte et piédroit) et deux images successives admettent un recouvrement de 95 % environ pour les tunnels canaux et 80 % pour les tunnels routiers.

Deux séquences issues respectivement d'un tunnel routier en béton et d'un tunnel navigable en maçonnerie ont été utilisées dans le cadre de nos expérimentations. Compte tenu des différences de nature des défauts recherchés, elles ont été traitées séparément. Concrètement, on s'intéresse ici à quatre types de défauts pour les revêtements en béton (fissures, zones humides, fers apparents, nids de cailloux). Seules les zones humides sont traitées pour les parements en maçonnerie . Pour pallier la rareté de certains défauts, on décide de simplifier le problème en classifiant les échantillons de façon binaire (défaut/sain).

3.2 Prétraitement

Afin de localiser les défauts, on propose de travailler sur des imagettes (ou échantillons) extraites des prises de vues au lieu de traiter des images entières. Les images originales sont partitionnées en imagettes de taille 101×101 pixels pour les parois en béton et 251×251 pour celles en maçonnerie. L'utilisation d'un plus grand format pour ces dernières est motivé par le fait qu'une trop petite taille d'imagette ne permettrait pas de recouvrir plusieurs moellons dans la même zone. La figure 3 montre quelques imagettes par type de revêtement et pour chaque catégorie. On peut observer l'hétérogénéité d'aspect des échantillons à l'intérieur de chaque classe, notamment pour les tunnels en béton.

La labellisation, également réalisée par nos soins, s'effectue en annotant chaque imagette en considérant qu'une

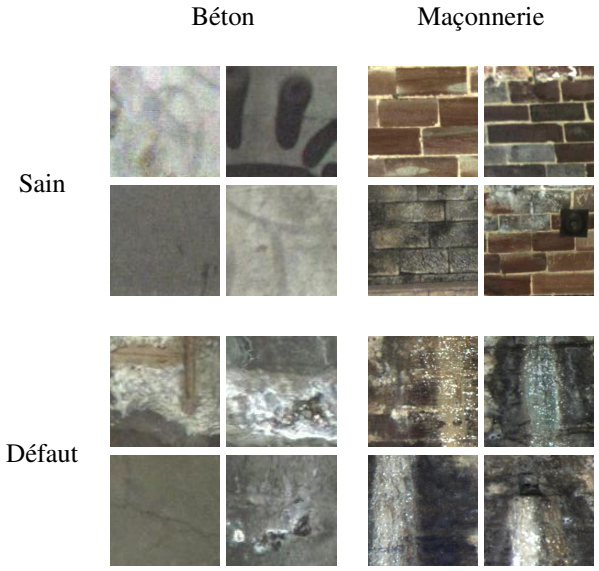


FIGURE 3 – Exemples d'imagettes de la base d'apprentissage ou de test. Pour les tunnels en béton, les imagettes de la classe défaut représentent de gauche à droite et haut en bas : un fer apparent, une arrivée d'eau, une fissure et un nid de cailloux (altération en surface). Pour les tunnels en maçonnerie, la classe défaut ne comprend que des arrivées d'eau.

zone est déclarée « défaut » à partir du moment où elle contient (au moins) un défaut, même si ce dernier se situe au bord de l'imagette. On peut d'emblée faire le constat suivant : l'opération de labellisation est difficile puisqu'il n'est pas toujours évident de déterminer si la zone représentée par l'imagette contient un défaut sans avoir connaissance de son voisinage.

Une fois labellisées, les imagettes sont ensuite réparties au hasard entre les jeux d'apprentissage et de test. La table 1 détaille la composition des deux bases de travail ainsi obtenues.

3.3 Algorithmes de classification

Forêt aléatoire. L'algorithme de forêt aléatoire [2] est un méta-classifieur, constitué d'un ensemble d'arbres de décision. Dans cette approche par comité, chaque arbre classe un échantillon, représenté par son vecteur d'attributs, dans une catégorie. La classification finale de l'échantillon par la forêt est obtenue à partir du vote majoritaire issu de l'ensemble des arbres. Pour que les votes de chaque arbre soient indépendants, un jeu de données et d'attributs différent est présenté à chaque arbre lors de sa construction. Ces sous-ensembles de données et d'attributs sont extraits de manière aléatoire, avec remise, respectivement de l'ensemble des échantillons dédiés à l'apprentissage et de l'ensemble des caractéristiques utilisées. La sélection des données est effectuée à l'échelle de l'arbre tandis qu'un tirage des caractéristiques a lieu pour chaque noeud. Dans notre application, le nombre d'arbres de la forêt a été fixé de ma-

nière empirique à 200 et le nombre minimum d'éléments dans chaque feuille, noeud terminal de l'arbre, est fixé à 1. Les descripteurs choisis doivent représenter au mieux les caractéristiques observées sur les échantillons de chaque classe. Dans le cas des images de tunnel, trois types d'attributs ont été utilisés. Les attributs d'Haralick [7], calculés dans 4 orientations de la matrice de co-occurrence, apportent des informations de texture. Ces caractéristiques ont été adaptées aux images couleurs. Les histogrammes de gradients orientés (HOG) [4] mettent en évidence la distribution de l'orientation des contours dans l'image (en niveau de gris). Enfin les relations binaires entre un pixel et ses 8 voisins sont extraites grâce à la transformée Census [18]. La distribution statistique des valeurs Census est ensuite calculée pour former le descripteur mCentrist [15], dans lequel la couleur est également prise en compte. Au final, on regroupe toutes les caractéristiques dans un seul vecteur de dimension 3161 avec 95 valeurs pour les attributs d'Haralick, 18 composantes pour les HOG, et 3048 valeurs dans le descripteur mCentrist.

Réseau convolutif. Quatre architectures ont été utilisées, deux pour les parements en béton et deux autres pour ceux en maçonnerie. Pour chaque type de revêtement, un réseau « superficiel » (basé sur LeNet [10]) et un réseau « profond » (reprenant la structure de VGG19 [13]) ont été testés. Pour le réseau superficiel, le modèle a été choisi pour sa rapidité d'apprentissage et l'architecture du réseau profond a été retenue car elle nous semblait représenter un bon compromis entre expressivité du modèle et puissance de calcul nécessaire à l'entraînement. Pour tous les réseaux, le nombre de neurones dans le perceptron multicouche a été ajusté en fonction de la taille des images et du nombre de classes, afin de conserver le même niveau d'expressivité. De plus, le réseau superficiel pour la maçonnerie a été enrichi de plusieurs couches de convolution pour mieux rendre compte de la complexité du type du parement en comparaison du béton, qui présente une surface saine majoritairement lisse. Ces architectures sont détaillées dans la figure 4. Elles n'ont pas été pré-entraînées, nous les avons apprises de bout en bout.

Avant la phase d'apprentissage, les imagettes sont normalisées en envoyant chaque pixel de façon linéaire dans le cube $[-1;1]^3$ par l'application de $x \mapsto \frac{2}{255}x - 1$ sur chaque composante.

L'algorithme utilisé est la descente de gradient stochastique (avec l'entropie croisée pour critère d'optimisation) dont le pas d'apprentissage l_i de l'époque $i \in \mathbb{N}^*$ est fixé à

$$l_i = \frac{l_0}{1 + d \times i}$$

où les paramètres l_0 et d ont été empiriquement fixés à 0.01 et 0.3 pour le béton et 0.01 et 1 pour la maçonnerie.

Le nombre d'époques effectuées dépend de l'architecture (voir table 2). Initialement fixé à 250, ce nombre a été ajusté à la hausse ou à la baisse suivant le stade de l'apprentissage. Par exemple, le réseau superficiel pour le béton a convergé plus rapidement que le réseau profond sur la maçonnerie. Le premier a convergé plus tôt, tandis que le second a été interrompu avant d'avoir fini de converger.

Entrée : $(101, 101, 3)$
conv5-6
avgpool-2
conv5-16
avgpool-2
conv1-120
fc-84
dropout (0.5)
fc-2
Softmax

(a) Réseau superficiel (béton)

Entrée : $(251, 251, 3)$
conv5-8
avgpool-2
conv5-16
avgpool-2
conv5-32
avgpool-2
conv5-64
avgpool-2
fc-512
dropout (0.5)
fc-512
dropout (0.5)
fc-2
Softmax

(b) Réseau superficiel (maçonnerie)

Entrée : $(101, 101, 3)$ ou $(251, 251, 3)$
Couches conv. identiques à VGG19 [13]
fc-32
fc-32
fc-2
Softmax

(c) Réseaux profonds (béton et maçonnerie)

conv$k$**-**m	couche de convolution à m filtres dont le noyau est de dimension $k \times k$
avgpool-k	couche de pooling de fenêtre $k \times k$
fc-n	perceptron multicouche à n neurones
dropout (p)	dropout avec une probabilité de p

FIGURE 4 – Présentation des architectures convolutives (la convention utilisée pour les notations est celle de [13])

3.4 Algorithmes de détection et de segmentation

Les différents classifieurs ont été entraînés et testés sur la base de données précédemment décrite. Pour la phase de détection, on réemploie ces classifieurs au sein d'un algorithme de fenêtre glissante sur des images entières (non présentes dans les jeux d'apprentissage et de test). Concrètement, un algorithme de fenêtre glissante sans recouvrement est appliqué en partant du coin supérieur gauche de l'image. La taille de la fenêtre est identique à celle des imagettes. Ainsi, certains pixels sur le bas et la droite de l'image ne sont pas pris en compte. On obtient alors une segmentation par bloc de l'image, dans laquelle les défauts sont mis en surbrillance.

Pour les réseaux de neurones, une approche pour réali-

Modèle	Réseau superficiel	Réseau profond
Béton	181	250
Maçonnerie	250	912

TABLE 2 – Nombre d'époques

ser une segmentation a également été tentée. S'appuyant également sur le principe de la fenêtre glissante, il s'agit, cette fois-ci, de réaliser un parcours exhaustif de la fenêtre sur toute l'image, au pas du pixel. La probabilité d'appartenance à l'une des deux catégories recherchées est dans un premier temps calculée par fenêtre et associée à chaque pixel de cette fenêtre. En moyennant, pour chaque pixel, l'ensemble des prédictions associées aux fenêtre auxquelles il appartient, on obtient la probabilité d'appartenance du pixel à l'une des deux classes. A partir de celle-ci, l'image est segmentée en deux classes.

4 Résultats

4.1 Classification

Pour mesurer les performances des différents algorithmes de classification, on utilise l'exactitude, la précision et le rappel sur le jeu de test. L'exactitude correspond à la proportion d'exemples bien classés. Formellement, si M est la matrice de confusion associée au modèle sur le jeu de test, on a la relation suivante :

$$\text{exactitude} = \frac{\text{Tr}(M)}{\|M\|_1}$$

De manière classique, la précision est définie par la proportion d'échantillons de la catégorie défauts classés de manière correcte dans cette classe. Le rappel est la proportion d'échantillons de la catégorie défauts classés dans cette classe parmi tous les échantillons défauts à identifier. Pour notre contexte applicatif, on cherche à maximiser l'exactitude en privilégiant le rappel sur la précision. Ainsi, on préfère avoir quelques fausses alarmes qu'omettre des anomalies.

Les résultats sont reportés dans la figure 5. On remarque que, sur le béton, les réseaux convolutifs réussissent mieux que les forêts aléatoires à détecter les anomalies, indépendamment de la profondeur de l'architecture. On peut également voir que les valeurs pour la précision et le rappel sont significativement meilleures pour les réseaux convolutifs que pour la forêt aléatoire. De plus, ces valeurs sont mieux équilibrées pour les réseaux de neurones.

Pour la maçonnerie, le réseau superficiel et la forêt aléatoire atteignent des exactitudes similaires (avec un léger avantage pour le premier modèle). Le réseau profond a davantage de difficulté à apprendre des caractéristiques discriminantes. Son exactitude est significativement moins élevée en comparaison des deux autres modèles. Comme mentionné en section 3.3, l'apprentissage de ce modèle a été interrompu avant que ce dernier n'ait pu converger.

Quant à la précision et au rappel, on peut noter que le réseau superficiel devance la forêt aléatoire. Il n'en va pas de même pour le réseau profond, dont le rappel est clairement en-dessus. Cette idée transparaît très clairement dans la matrice de confusion (cf. figure 5f) où 71 imagettes ont été classifiées à tort comme revêtement sain. Néanmoins, c'est ce dernier qui obtient la meilleure précision de tous les modèles testés sur la maçonnerie.

$$\begin{array}{cc} & \begin{matrix}\text{S} & \text{D}\end{matrix} \\ \begin{matrix}\text{S}\\ \text{D}\end{matrix} & \begin{pmatrix} 71 & 6 \\ 17 & 184 \end{pmatrix} \end{array}$$

(a) Forêt aléatoire (béton)

$$\begin{array}{cc} & \begin{matrix}\text{S} & \text{D}\end{matrix} \\ \begin{matrix}\text{S}\\ \text{D}\end{matrix} & \begin{pmatrix} 284 & 18 \\ 23 & 245 \end{pmatrix} \end{array}$$

(b) Forêt aléatoire (maçonnerie)

$$\begin{array}{cc} & \begin{matrix}\text{S} & \text{D}\end{matrix} \\ \begin{matrix}\text{S}\\ \text{D}\end{matrix} & \begin{pmatrix} 73 & 4 \\ 5 & 196 \end{pmatrix} \end{array}$$

(c) Réseau superficiel (béton)

$$\begin{array}{cc} & \begin{matrix}\text{S} & \text{D}\end{matrix} \\ \begin{matrix}\text{S}\\ \text{D}\end{matrix} & \begin{pmatrix} 278 & 24 \\ 12 & 256 \end{pmatrix} \end{array}$$

(d) Réseau superficiel (maçonnerie)

$$\begin{array}{cc} & \begin{matrix}\text{S} & \text{D}\end{matrix} \\ \begin{matrix}\text{S}\\ \text{D}\end{matrix} & \begin{pmatrix} 73 & 4 \\ 5 & 196 \end{pmatrix} \end{array}$$

(e) Réseau profond (béton)

$$\begin{array}{cc} & \begin{matrix}\text{S} & \text{D}\end{matrix} \\ \begin{matrix}\text{S}\\ \text{D}\end{matrix} & \begin{pmatrix} 294 & 8 \\ 71 & 197 \end{pmatrix} \end{array}$$

(f) Réseau profond (maçonnerie)

Matrices de confusion des différentes modèles (S : sain, D : défaut, les vérités-terrain sont en ligne et les catégories détectées en colonne)

Forêt aléatoire	Réseau superficiel	Réseau profond
91,73%	**96.76%**	**96.76%**
92,81%	**93.68%**	86.14%

Exactitudes obtenues par les différents modèles.

Forêt aléatoire	Réseau superficiel	Réseau profond
(96.84, 91.54)	(98.00, 97.51)	(98.00, 97.51)
(93.16, 91.42)	(91.43, 95.52)	(96.10, 73.51)

Valeurs (en %) de précision/rappel pour les trois modèles.

FIGURE 5 – Résultats de l'expérimentation. Pour les tableaux donnant l'exactitude et le couple précision/rappel, les résultats pour le béton sont sur la première ligne et ceux de la maçonnerie sur la deuxième.

Dans la figure 6, on montre quelques exemples d'imagettes mal classées. Les imagettes 6b et 6d ont été classées comme « Saines » (vérité-terrain : Défaut). Pour ces exemples, on peut noter que les défauts en question sont situés sur le bord de l'imagette et ils sont difficilement identifiables. Les exemples 6a et 6c sont considérés comme des défauts par les algorithmes. Pour le premier, la présence d'un tag peut expliquer cette erreur. Quant au second, la

(a) Sain (b) Défaut

Béton

(c) Sain (d) Défaut

Maçonnerie

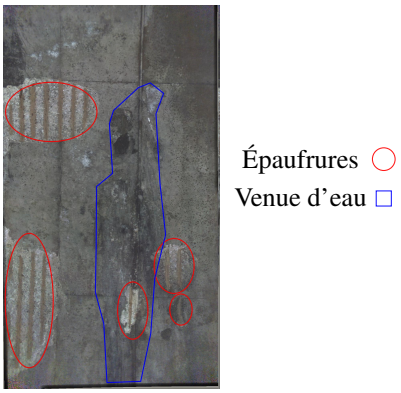
FIGURE 6 – Exemples d'imagettes simultanément mal classées par les deux réseaux de neurones. Les exemples de la première colonne ont été classés à tort parmi les défauts et inversement pour ceux de la seconde. Le titre correspond au label de la vérité-terrain associée à l'exemple.

variabilité d'aspect dans le moellon peut avoir suffi à tromper les réseaux de neurones.

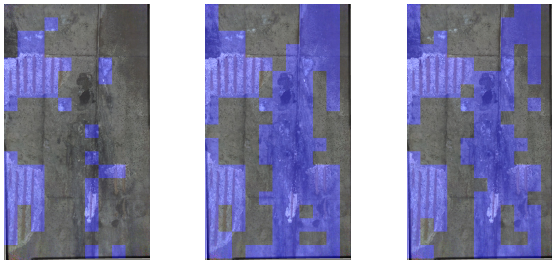
4.2 Détection et segmentation

Pour le moment, nous ne disposons pas d'une vérité terrain suffisamment précise pour pouvoir évaluer de façon quantitative la détection. On présente dans la figure 7 le résultat d'une détection sur un exemple d'image de parement en béton. On remarque que les trois modèles fournissent des résultats corrects pour les épaufrures. En revanche, la venue d'eau est mal détectée par la forêt aléatoire alors qu'elle est correctement identifiée par les réseaux de neurones, au prix d'un plus grand nombre de faux positifs. Le fort taux de fausses alarmes peut s'expliquer par la petitesse de notre jeu d'apprentissage. En effet, les réseaux de neurones sont réputés pour avoir besoin de beaucoup de données pour être appris de façon efficace. Nous pouvons également remarquer que les deux réseaux de neurones produisent des résultats proches mais distincts, alors que leur matrices de confusion sont identiques.

La figure 8 présente le résultat d'une segmentation par les réseaux de neurones sur la même image. On peut, en premier lieu, dresser le constat rassurant qu'ils sont en adéquation avec les résultats de détection qui leur sont associés. Plus généralement, il en ressort le lissage des scores par moyennage ne suffit pas à réduire le nombre de fausses alarmes, même si le réseau profond semble être un peu plus performant sur ce point. En dépit de cette proportion élevée de fausses alarmes, certains défauts demeurent mal détectés comme l'épaufrure en bas à gauche de l'image qui présente systématiquement une zone non reconnue par les réseaux de neurones.

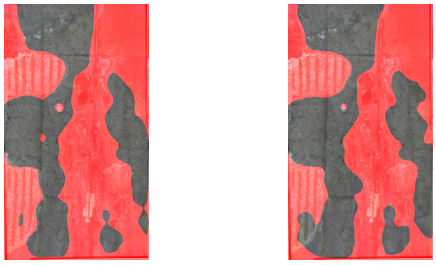


(a) Image de test labellisée à la main



(b) (c) (d)

FIGURE 7 – Détection des anomalies sur un exemple de parement en béton (les anomalies apparaissent en bleu) avec l'algorithme de forêt aléatoire (b), le réseau superficiel (c) et le réseau profond (d).



(a) (b)

FIGURE 8 – Segmentation des anomalies sur le même exemple qu'à la figure 7 avec le réseau superficiel (b) et le réseau profond (c). Les anomalies apparaissent en rouge.

5 Conclusion et perspectives

Les méthodes d'apprentissage proposées nous ont permis, dans un cadre certes limité (taille de la base d'apprentissage et de test, classification binaire) d'obtenir des performances comparables ou supérieures à l'état de l'art dans le domaine. ainsi, pour les tunnels en béton, les trois ap-

proches proposées ont une exactitude supérieure à 90%, obtenue sur une base de taille limitée. Pour les revêtements en maçonnerie, les résultats sont un peu plus contrastés, notamment pour le réseau basé sur une architecture VGG. Le nombre relativement réduit d'échantillons peut expliquer les difficultés d'apprentissage de cette architecture profonde. Pour conclure plus avant sur ce point, il conviendra toutefois de comparer nos modèles avec ceux de la littérature, sur des bases communes.
Au vu des résultats obtenus en détection/segmentation, les réseaux de neurones semblent générer beaucoup de fausses alarmes tandis que les forêts aléatoires détectent plus difficilement les défauts.

L'introduction de modèles plus complexes, associés à des bases d'apprentissage significativement étoffées, pourrait permettre de répondre à plusieurs limitations de notre implantation actuelle.
Le premier point est la représentativité limitée des données traitées, issues d'un petit nombre de tunnels, avec un nombre d'exemples limité à un millier pour chaque type de revêtement. Les données actuellement disponibles ne sont donc que partiellement représentatives des relevés qui pourraient être effectués dans d'autres tunnels ou sur des structures mettant en jeu d'autres matériaux. Avec une base d'apprentissage aussi restreinte, on peut donc s'attendre à une faible capacité de généralisation de nos réseaux sur des structures non observées.
Une seconde limitation est la classification en deux classes seulement, alors qu'il serait souhaitable d'obtenir une caractérisation plus fine des différents types de défaut ou du niveau de gravité de l'anomalie détectée.
Un troisième point d'amélioration est d'aller vers une localisation plus précise des anomalies, en évitant de recourir à une méthode de fenêtre glissante, coûteuse en temps de calcul, pour aboutir à une segmentation directe des défauts. On pourra utiliser, à cet effet, des architectures convolutives dédiées à cette tâche telles Mask R-CNN [8] ou RetinaNet [11].
Pour répondre à ces différents points, une priorité sera l'augmentation des bases de données d'apprentissage, en diversifiant les acquisitions sur d'autres tunnels, en exploitant les redondances temporelles sur les séquences acquises (un défaut étant vu de plusieurs points de vue) et en recourant à des méthodes d'augmentation de données (de nouvelles campagnes d'acquisition sont d'ores et déjà programmées). Il conviendra également de retravailler sur les modèles profonds pour augmenter leur capacité de représentation, tout en évitant le sur-apprentissage. Enfin on s'appuiera sur différentes stratégies pour effectuer des transferts d'apprentissage à partir de modèles préentraînés, en s'adaptant à la nature et au volume des nouvelles données collectées [17].

Références

[1] AMHAZ, R., CHAMBON, S., IDIER, J., AND BALTAZART, V. Automatic Crack Detection on Two-Dimensional Pavement Images : An Algorithm Based on Minimal Path Selection. *IEEE Transactions on Intelligent Transportation Systems 17*, 10 (2016), 2718–2729.

[2] BREIMAN, L. Random Forests. *Machine Learning 45*, 1 (2001), 5–32.

[3] CHAIYASARN, K., KHAN, W., ALI, L., SHARMA, M., BRACKENBURY, D., AND DEJONG, M. Crack Detection in Masonry Structures using Convolutional Neural Networks and Support Vector Machines. *2018 Proceedings of the 35th ISARC* (2018), 118–125.

[4] DALAL, N., AND TRIGGS, B. Histograms of oriented gradients for human detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) 1* (2005), 886–893.

[5] FENG, C., LIU, M.-Y., KAO, C.-C., AND LEE, T.-Y. Deep Active Learning for Civil Infrastructure Defect Detection and Classification. *Computing in Civil Engineering 2017* (2017), 298–306.

[6] FOUCHER, P., BAH, M. D., CHARBONNIER, P., BOULOGNE, C., AND LARIVÉ, C. Classification automatique de défauts sur des images de tunnels par forêts d'arbres aléatoires. *Congrès national sur la Reconnaissance de Formes et l'Intelligence Artificielle (RFIA)* (2016), 1–2.

[7] HARALICK, R., SHANMUGAM, K., AND DINSTEIN, I. Textural Features for Image Classification. *IEEE Transactions on Systems, Man, and Cybernetics SMC-3*, 6 (1973), 610–621.

[8] HE, K., GKIOXARI, G., DOLLÁR, P., AND GIRSHICK, R. Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2018), 1–12.

[9] HE, K., ZHANG, X., REN, S., AND SUN, J. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 770–778.

[10] LECUN, Y. Generalization and network design strategies. In *Connectionism in perspective*, elsevier ed. R. Pfeifer and Z. Schreter and F. Fogelman and L. Steels, 1989, pp. 1–20.

[11] LIN, T.-Y., GOYAL, P., GIRSHICK, R., HE, K., AND DOLLÁR, P. Focal Loss for Dense Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2018), 1–10.

[12] MAKANTASIS, K., PROTOPAPADAKIS, E., DOULAMIS, A., DOULAMIS, N., AND LOUPOS, C. Deep Convolutional Neural Networks for efficient vision based tunnel inspection. *2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)* (2015), 335–342.

[13] SIMONYAN, K., AND ZISSERMAN, A. Very Deep Convolutional Networks For Large-Scale Image Recognition. *Proceedings of ICLR 2015* (2015), 1–14.

[14] WANG, P., AND HUANG, H. Comparison analysis on present image-based crack detection methods in concrete structures. *2010 3rd International Congress on Image and Signal Processing 5* (2010), 2530–2533.

[15] XIAO, Y., WU, J., AND YUAN, J. mCENTRIST : A Multi-Channel Feature Generation Mechanism for Scene Categorization. *IEEE Transactions on Image Processing 23*, 2 (2014), 823–836.

[16] YAMAGUCHI, T., NAKAMURA, S., AND HASHIMOTO, S. An efficient crack detection method using percolation-based image processing. *2008 3rd IEEE Conference on Industrial Electronics and Applications* (2008), 1875–1880.

[17] YOSINKI, J., CLUNE, J., BENGIO, Y., AND LISPON, H. How Transferable are Features in Deep Neural Networks ? *Proceedings of the 27th International Conference on Neural Information Processing Systems 2* (2014), 3320–3328.

[18] ZABIH, R., AND WOODFILL, J. Non-parametric local transforms for computing visual correspondence. In *European Conference on Computer Vision (ECCV '94)* (Berlin, Heidelberg, 1994), Springer Berlin Heidelberg, pp. 151–158.