

PROJET DE DÉVELOPPEMENT INFORMATIQUE

ANNÉE 2023-2024

ING22

Rapport d'analyse

« *Détecter le monstre des marais dans les cartes anciennes de l'IGN* »

— GROUPE PDI 09 —

Lilia CAMPO
Suzanne LIZOT
Vanessa PECH
Elisa PEYRARD

— *Commanditaires* —

Guillaume TOUYA, *guillaume.touya@ign.fr*
Hervé QUINQUENEL, *herve.quinquenel@ign.fr*
Azelle COURTIAL, *azelle.courtial@ign.fr*

Sommaire

1 Contexte du projet	2
1.1 Commanditaires	2
1.2 Détection d'un problème avec les cartes actuelles de l'IGN	2
2 Objectifs de l'étude	3
2.1 Les objectifs de l'étude	3
2.2 Les contraintes	3
2.3 Le recueil du besoin - Les acteurs	4
3 Analyse fonctionnelle - Solutions proposées	5
3.1 Fonctionnalités principales identifiées	5
3.2 Propositions et évolution de la modélisation	6
4 Etude technique de notre projet	6
4.1 Données à disposition	6
4.2 1ère étape : Création des cartes anciennes	6
4.3 2ème étape : Entraînement du modèle de deep learning	7
4.3.1 Faire un masque de pictogrammes pour une image donnée	7
4.3.2 Générer automatiquement beaucoup d'images	8
4.3.3 Implémenter un code pour entraîner un modèle référence pour des problèmes de reconnaissance de formes.	8
4.3.4 Résultats des tests sur de véritables cartes anciennes.	11
5 Réalisation et suivi de projet	12
5.1 Choix du nom de notre projet	12
5.2 Logo	13
5.3 Organisation	13

5.4	Planning prévisionnel	13
5.5	Distribution des tâches	14
5.6	Les risques et difficultés rencontrées	14
6	Conclusion	15
7	Annexes	16

1 Contexte du projet

1.1 Commanditaires

Les commanditaires de notre projet sont Hervé Quinquenel, responsable de l'équipe Produit Cartographies et Chaîne Graphique, Azelle Courtial, post-doctorante, et Guillaume Touya, directeur de recherche. Tous les trois sont des agents de l'Institut national de l'information géographique et forestière (IGN), un établissement public relevant des ministères chargés de l'écologie et de la forêt. L'IGN a pour mission entre autres la production et la diffusion de données ainsi que de cartes topographiques.

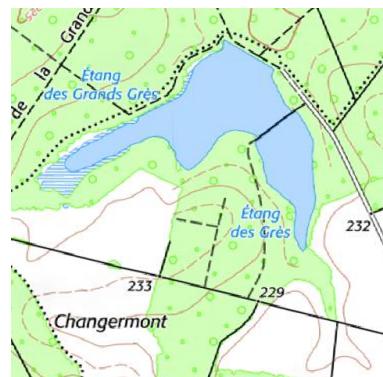
1.2 Détection d'un problème avec les cartes actuelles de l'IGN

Les cartes topographiques anciennes de l'IGN contiennent des détails concernant des thèmes qui ne sont plus entretenus de façon qualitative en raison de contraintes de temps ou de priorités. Les cartes de l'IGN étant produite à partir de la BDTopo, les cartographes enrichissent parfois cette base de données manuellement ces cartes actuelles en exploitant ces données anciennes mais c'est donc un travail fastidieux qui leur prend beaucoup de temps, et qui est censé être réalisé par le producteur de la BDTopo.

Nos commanditaires nous ont ainsi demandé de nous concentrer spécifiquement sur les pictogrammes représentant les marais. En effet, en comparant une carte récente figure 1(a) avec une carte plus ancienne figure 1(b), on peut remarquer que certains marais ne sont plus indiqués sur les cartes actuelles, alors qu'ils sont toujours présents sur le terrain. Notamment les pictogrammes présents dans les forêts.



(a) Carte ancienne possédant des pictogrammes de marais



(b) Carte récente ne possédant plus ces pictogrammes

FIGURE 1 – Exemple de perte d'information des pictogrammes de marais

2 Objectifs de l'étude

2.1 Les objectifs de l'étude

Le but principal de notre projet est donc de détecter les pictogrammes des marais sur les cartes anciennes afin de les localiser précisément et ainsi de les rajouter dans la Base de Données Topographique (BD Topo). Une fois dans la BD topo, on pourra les rajouter sur les cartes actuelles et ainsi résoudre le problème de la disparition des marais sur les cartes récentes. Cependant, il ne faut pas oublier que les pictogrammes ne sont qu'une représentation cartographique destinés à la carte. Dans la BDtopo, la modélisation des zones humides est surfacique. Détecter des pictogrammes pourraient être une alerte pour les bases d'OCS (base de données de référence pour la description de l'occupation du sol de l'ensemble du territoire métropolitain et des départements et régions d'outre-mer) qui seront ainsi intégrées ensuite éventuellement dans la BDTopo.

L'objectif principal de notre application est de développer un algorithme capable de détecter précisément les pictogrammes de marais sur des portions de cartes anciennes.

Pour cela, l'architecture technique de la solution serait de concevoir une chaîne de traitement d'images pour prétraiter les données cartographiques avant la détection des pictogrammes. Ce pré-traitement à pour objectif de constituer un jeu d'apprentissage pour le modèle d'apprentissage développé ensuite. Puis d'élaborer un modèle de détection de pictogrammes efficace et précis en utilisant des techniques de reconnaissance de formes et d'apprentissage automatique. En ce qui concerne le respect des performances imposées par le cahier des charges de nos commanditaires, on pourra évaluer la précision de l'algorithme de détection en utilisant des jeux de données variés représentatifs des cartes anciennes. Ce projet permettrait ainsi d'apporter une valeur ajoutée à l'IGN en améliorant la précision et l'exhaustivité de ses cartes, sa BD topographiques et d'OCS.

2.2 Les contraintes

Dans le cadre de ce projet, les contraintes imposées par le commanditaire sont principalement axées sur les choix des algorithmes de deep learning pour la détection des pictogrammes de marais sur les cartes topographiques.

Le commanditaire a expressément spécifié que la solution de détection des pictogrammes de marais doit être basée sur l'utilisation des algorithmes UNet ou SegFormer pour le deep learning. En dehors de cette contrainte spécifique concernant les algorithmes de deep learning, le commanditaire n'a imposé aucune autre restriction quant aux choix technologiques ou aux outils à utiliser pour le développement de la solution.

Il convient de noter que les contraintes de temps sont également un aspect crucial à prendre en compte. Les délais impartis pour la réalisation du projet ont été discutés avec le commanditaire et sont intégrés à la planification globale du projet.

2.3 Le recueil du besoin - Les acteurs

Nous avons identifié les différents acteurs impliqués dans le projet de détection des pictogrammes de marais sur les cartes anciennes de l'IGN. Les acteurs comprennent les commanditaires du projet, tels que Hervé Quinquenel, responsable de l'équipe Produit Cartographies et Chaîne Graphique, Azelle Courtial, post-doctorante, et Guillaume Touya, directeur de recherche à l'IGN. Nous pouvons également citer le personnel de l'IGN chargé de la cartographie et de la mise à jour des données topographiques.

Le besoin identifié pour les commanditaires du projet est le besoin de disposer d'un système automatisé capable de détecter les pictogrammes de marais sur les cartes topographiques anciennes. Pour représenter ces besoins, nous avons créé un diagramme de cas d'utilisation.

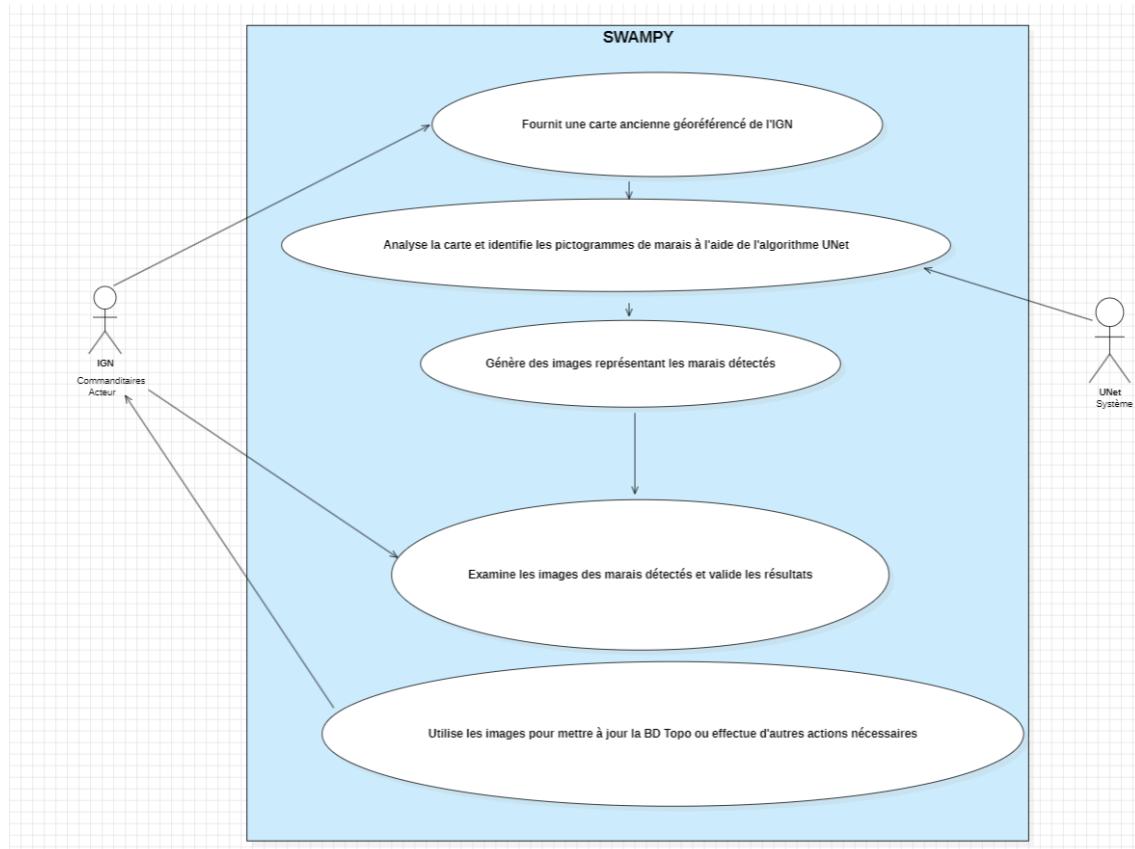


FIGURE 2 – Diagramme des cas d'utilisation de notre application

3 Analyse fonctionnelle - Solutions proposées

3.1 Fonctionnalités principales identifiées

1. *Création des cartes anciennes simulées*

Cette fonctionnalité consiste à utiliser le logiciel QGIS pour créer des cartes anciennes à partir des données disponibles actuelles, en reproduisant le style des cartes topographiques anciennes de l'IGN. Cela implique la modification du style des différentes couches de la carte pour les rendre conformes aux cartes anciennes, ainsi que la création et l'ajout de pictogrammes de marais sur ces cartes. Le fait de générer nous-même la position des pictogrammes permet de faciliter la création du masque.

2. *Création du masque correspondant aux cartes anciennes simulées*

Cette fonctionnalité implique l'utilisation de QGIS pour générer des masques correspondant aux pictogrammes de marais sur les cartes anciennes simulées. Pour ce faire, une grille noire est créée pour couvrir toute la surface de la carte ancienne simulée. Ensuite, les pictogrammes de marais sont superposés sur cette grille aux endroits correspondant à leur emplacement sur la carte ancienne. Cette superposition crée des masques binaires où les zones de marais sont représentées par des pixels blancs et le reste de la grille est noir.

3. *Entraînement du modèle de deep learning*

Cette fonctionnalité vise à entraîner un modèle de deep learning (tel que UNet ou SegFormer) pour détecter les pictogrammes de marais sur les cartes anciennes. Elle comprend la constitution d'un jeu de données d'entraînement à partir des cartes anciennes créées, la préparation des données pour l'entraînement du modèle, le choix et l'entraînement du modèle, ainsi que l'évaluation de sa performance.

4. *Découpage des vraies cartes anciennes en portions qui peuvent être traitées par la modèle d'apprentissage*

Cette fonctionnalité consiste à découper les vraies cartes anciennes en portions plus petites, appelées tuiles, qui peuvent être traitées par le modèle d'apprentissage en raison de leur taille. Ce découpage peut être effectué manuellement ou automatiquement en utilisant des algorithmes de segmentation d'image. Une fois les cartes anciennes découpées en tuiles, chaque tuile peut être traitée individuellement par le modèle d'apprentissage pour la détection des pictogrammes de marais.

3.2 Propositions et évolution de la modélisation

Il est important de noter que la modélisation proposée peut évoluer au cours de l'implémentation en fonction des besoins et des contraintes rencontrés. Par exemple, si une fonctionnalité s'avère plus complexe que prévu, elle pourrait nécessiter des ajustements dans le modèle initial. Il est donc essentiel de rester flexible et de communiquer efficacement avec les commanditaires pour justifier les changements et assurer le succès du projet.

4 Etude technique de notre projet

4.1 Données à disposition

- ❖ Une couche surfacique de la carte actuelle correspondant aux végétations aquatiques de la carte 1 : 25000 de l'IGN en fichier shapefile.(cf figure 9)
- ❖ Des cartes anciennes 1 : 25000 non géoréférencées (la Carte TOP 25 n° 3331 OT datant de 1997, la carte 2714 E datant de 2006 et la Carte TOP 25 n° 3331 OT datant de 2007) en fichier JP2. (cf figure 8)
- ❖ 15 couches vectorielles correspondant aux couches de la BD Topo (Bois, Bassin, cours d'eau temporaire, ect) du département des Bouches du Rhône 1 : 25000 de l'IGN en fichier shapefile.
- ❖ 15 couches vectorielles correspondant aux couches de la BD Topo (Bois, Bassin, cours d'eau temporaire, ect) appartenant à la zone proche de l'intersection Oise/Seine-et-Marne/Aisne : 25000 de l'IGN en fichier shapefile.

4.2 1ère étape : Crédation des cartes anciennes

Pour pouvoir entraîner notre modèle, nous avons besoin de créer nos propres cartes anciennes, par manque de données, nous utilisons donc les données actuelles que nous avons à disposition pour créer ces cartes anciennes simulées. Pour réaliser cela, nous avons utilisé le logiciel QGIS, un logiciel gratuit et que nous maîtrisons grâce à notre cursus à l'ENSG. Ainsi, à partir de la BD topographique et des 15 couches vectorielles que nous a fourni Hervé, nous avons modifié le style de ces couches afin de reproduire au mieux le style carte ancienne.

Toujours en s'inspirant des 3 cartes anciennes que nous avons, nous reproduisons les pictogrammes de marais à l'aide d'outils comme paint, que nous enregistrons sous format SVG, afin de pouvoir les utiliser comme style sur la couche correspondant au marais sur QGIS. Nous avons ainsi 3 pictogrammes de la Carte TOP 25 n° 3331

OT datant de 1997 (cf figure 11), 3 pictogrammes de la carte 2714 E datant de 2006 (cf figure 12) et 3 pictogrammes de la Carte TOP 25 n° 3331 OT datant de 2007 (cf figure 13).

Nous avons ainsi reproduit des cartes ressemblant à des cartes anciennes. Un premier style correspondant aux Cartes TOP 25 n° 3331 OT .(figure 14 ou figure 16) La seule chose qui change selon l'année de la carte TOP 25 est le style du pictogramme des marais.(voir la légende ??).Le deuxième correspond donc au style de la carte 2714 E ??). La méthode que nous avons utilisé pour réaliser le style 2714 E se trouve sur figure 18.

4.3 2ème étape : Entraînement du modèle de deep learning

Une fois les cartes anciennes créées, nous pouvons passer à la deuxième étape. Pour entraîner un modèle de deep learning à reconnaître des pictogrammes, il faut construire un jeu d'entraînement d'images c'est-à-dire constituer un dossier de milliers d'images png contenant : des images modèles (portion de carte avec ou sans marais) et des images du masque associée aux images modèles (seulement les pictogrammes de marais).

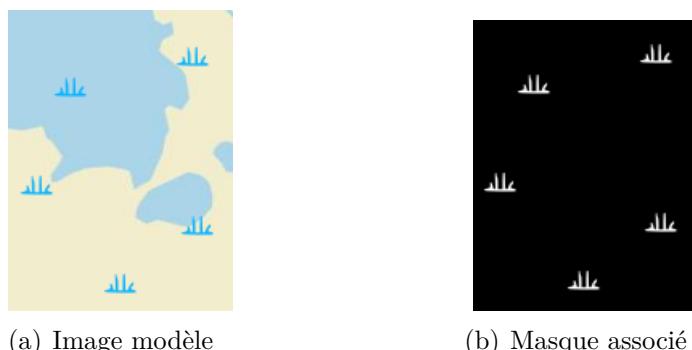


FIGURE 3 – Images constituant le jeu de donnée d'entraînement

4.3.1 Faire un masque de pictogrammes pour une image donnée

Nous avons mis en place un protocole QGIS permettant une première étape pour mettre en place le jeu de données :

- ❖ Créer un polygone de couleur noir et de la taille que vous souhaitez sur la zone étudiée.
- ❖ Couper le polygone avec la couche des marais (ici : *zone_veget_aquatique*).
- ❖ Utiliser la fonction « points aléatoires à l'intérieur des polygones » avec un

nombre de points de 1 et une distance entre chaque point de 200 mètres sur la couche découpée.

- ❖ Charger le style adéquat en qml pour votre marais (2006,2007,1997).
- ❖ Supprimer la couche découpée.
- ❖ Enregistrer la carte avec la fonction exporter la carte en format image. Les paramètres sont l'emprise (couche du polygone) et la résolution (300dpi). Enregistrer l'image qui est donc le masque de l'image modèle, dans le dossier du code qui permet de découper.
- ❖ Décocher la couche du polygone et refaire la manipulation précédente pour enregistrer l'image modèle.

On obtient les deux images de la figure 10(b).

Vous pouvez retrouver le code python (à exécuter dans la console python de QGIS) pour réaliser cette étape dans le dépôt git du projet : [code pour créer un masque dans QGIS](#).

4.3.2 Générer automatiquement beaucoup d'images

- ❖ Exécutez le code que nous avons réalisé permettant de découper les cartes en petites vignettes ainsi que de passer de la couleur au binaire pour le masque sur les cartes que nous avons récupéré lors de l'étape précédente. Le code python utilisé se trouve dans le dépôt git : [code pour découper les images](#).
- ❖ Il est important que le dataset soit organisé de manière à que les images aient des noms similaires dans des répertoires différents pour que les images soient directement lues par les bibliothèques d'apprentissage profond.

4.3.3 Implémenter un code pour entraîner un modèle référence pour des problèmes de reconnaissance de formes.

Nos commanditaires nous ont suggéré d'utiliser comme modèle de deep learning : U-net ou SegFormer.

Pour notre cas Segformer est plus pertinent que U-net mais plus compliqué à prendre en main car assez récent. SegFormer prend en compte le contexte autour de l'image recherchée, donc plus intéressant pour l'étude de cartes (plus les éléments sont proches, plus ils ont une relation entre eux). Cependant, en plus d'être plus compréhensible pour un début d'étude de deep learning, U-Net permet de résoudre

facilement les problèmes de segmentation d'image et est performant pour des datasets qui ne sont pas trop volumineux. C'est pour ces raisons que nous avons focalisé notre travail sur le modèle U-Net.

Fonctionnement du modèle U-Net

Le processus commence par la récupération du dataset conçu à partir des fausses cartes anciennes que nous avons préalablement créées pour ressembler à des cartes de 1997, 2006 et 2007. Les générateurs de données sont utilisés pour charger les images et masques du dataset en lots, les prétraiter et les fournir au modèle pour l'entraînement.

Le modèle U-Net utilisé est un réseau de neurones convolutifs (CNN) avec une architecture spécifique pour la détection d'objets. Cette architecture comprend des couches de convolution pour extraire les caractéristiques des images, suivies de couches de détection pour prédire la présence et la localisation de l'objet attendu. U-Net utilise la librairie de machine learning TensorFlow ainsi que Keras, interface Python pour les réseaux de neurones artificiels intégrée à cette dernière. Le modèle est entraîné sur les données préparées en ajustant ses poids à l'aide d'un algorithme d'optimisation tel que la descente de gradient stochastique. Pendant l'entraînement, le modèle apprend à détecter les pictogrammes de marais en minimisant une fonction de perte qui mesure la différence entre les prédictions du modèle et les véritables emplacements des pictogrammes sur les cartes anciennes.

Pour évaluer la performance du modèle, celui-ci renvoie à chaque fin d'epoch (passage dans les données d'entraînement) les valeurs de pertes d'entraînement et de validation ainsi que les valeurs de précision d'entraînement et de validation mesurées par la métrique IoU (Intersection over Union). L'IoU est calculée en divisant la surface d'intersection entre la case dans laquelle se trouve un pictogramme et la case que le modèle prédit, par l'union de ces deux cases. Ces valeurs s'accompagnent du résultat en images de la phase de validation de l'epoch, où le modèle génère un masque à partir d'une imagette sur laquelle il ne s'est pas entraîné (issue du dataset de validation).



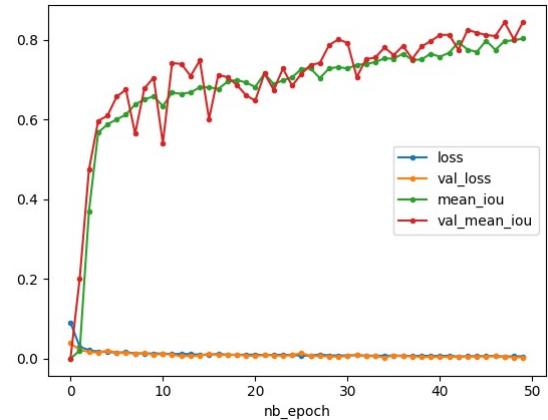
FIGURE 4 – Exemple d'affichage de sortie à la fin d'un epoch, avec batch size = 32 et epochs = 50

Les hyper-paramètres qui influent sur la performance du modèle et que nous faisons varier pour déterminer leurs valeurs optimales sont la taille d'un lot (ou batch) et le nombre d'epochs. Plus un lot est important, plus l'apprentissage est rapide puisque les mises à jour des poids qui se font à la fin d'un lot s'effectueront donc peu fréquemment. Cependant, de grands lots peuvent contenir plus de bruit et donc diminuer la précision du modèle. A l'inverse, de petits lots impliqueront un apprentissage lent et une meilleure précision. En augmentant le nombre d'epochs, on augmente le temps d'apprentissage du modèle mais aussi sa précision. Cependant, un nombre trop important d'epochs peut aussi engendrer un sur-apprentissage du modèle (le modèle saura très bien détecter les pictogrammes sur lesquels il s'est entraîné mais pas ceux qu'il découvrira pour la première fois, il ne saura donc pas généraliser).

Nous avons implémenté un code ([cf git](#)) qui nous permet, à la fin de l'entraînement, de visualiser et d'enregistrer un dataframe 5(a) contenant toutes les valeurs de performance citées précédemment ainsi qu'un graphique 5(b) afin de comparer aisément l'effet des hyper-paramètres sur la performance de U-Net. Nous avons aussi ajouté la possibilité de sauvegarder le modèle entraîné afin de le conserver et de le tester ultérieurement.

nb_epoch	loss	val_loss	mean_iou	val_mean_iou
0	0.090424	0.038545	0.000179	0.000000
1	1 0.029820	0.021717	0.020180	0.201139
2	2 0.021260	0.016570	0.369224	0.473762
3	3 0.017601	0.014903	0.567070	0.596386
4	4 0.017046	0.020049	0.587415	0.609121
5	5 0.014783	0.015110	0.500117	0.656917
6	6 0.016406	0.013523	0.611719	0.674831
7	7 0.013094	0.012965	0.637670	0.564196
8	8 0.013035	0.013673	0.649995	0.677714
9	9 0.012807	0.009641	0.657757	0.704554
10	10 0.012678	0.011650	0.633472	0.548126
11	11 0.011605	0.009989	0.667334	0.741256
12	12 0.011816	0.005848	0.663978	0.739036
13	13 0.011308	0.006811	0.667619	0.708927
14	14 0.009774	0.007503	0.680307	0.747508
15	15 0.009814	0.012276	0.580760	0.601075
16	16 0.011793	0.008859	0.676354	0.711730
17	17 0.009917	0.009538	0.694607	0.705733
18	18 0.009692	0.009262	0.699074	0.686663
19	19 0.009875	0.007558	0.692250	0.666675
20	20 0.010030	0.006884	0.681465	0.646941
21	21 0.009005	0.008737	0.719497	0.715116
22	22 0.009455	0.008209	0.688644	0.674139
...				

(a) Extrait du dataframe d'évaluation du modèle



(b) Graphique d'évaluation du modèle

FIGURE 5 – Fichiers récapitulatifs de l'évaluation du modèle, pour batch size = 32 et epochs = 50

Une fois l'entraînement terminé, on peut aborder la phase de test qui consiste simplement à fournir une image de véritable carte ancienne au modèle pour qu'il en déduise le masque associé et ainsi détecter les pictogrammes de marais.

En résumé, ce processus consiste à entraîner un modèle de détection d'objets à l'aide de données étiquetées, à ajuster les paramètres du modèle pour qu'il puisse détecter avec précision les pictogrammes de marais sur les cartes anciennes que nous avons créée, et à évaluer sa performance sur des données de test. Une fois entraîné, ce

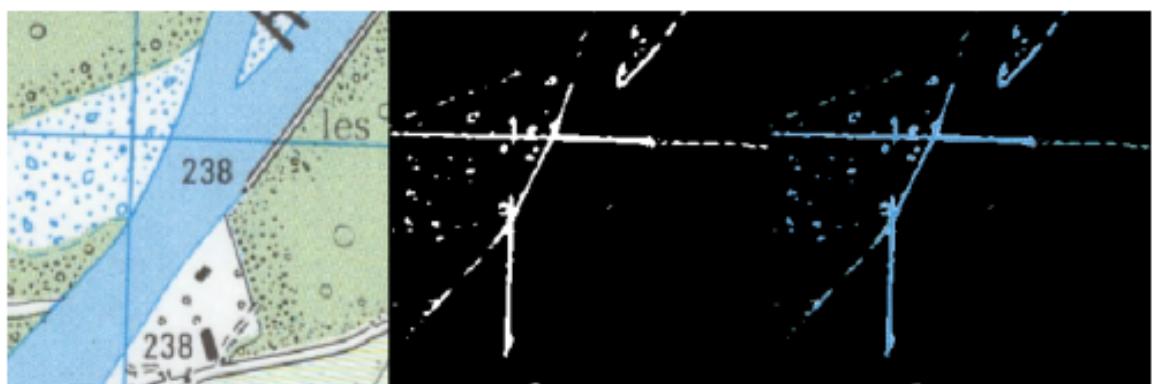
modèle peut être utilisé pour détecter automatiquement les pictogrammes de marais sur de vraies cartes anciennes.

4.3.4 Résultats des tests sur de véritables cartes anciennes.

Après avoir découpé de véritables cartes anciennes en imagettes de 256x256 pixels grâce au code qui nous permettait de découper les fausses cartes anciennes, nous en avons fourni quelques unes au modèle entraîné pour qu'il parvienne (ou non) à détecter des pictogrammes. Les imagettes choisies pour la phase de test contiennent, pour certaines, des pictogrammes, et pour d'autres, des objets que le modèle pourrait confondre avec des pictogrammes de marais (cf figure 19 et figure 21 en Annexe).



(a) Test sur une imagette issue d'une carte de 2006



(b) Test sur une imagette issue d'une carte de 1997

FIGURE 6 – Exemples de tests effectués, pour batch size = 32 et epochs = 30

Observations

Nous avons pu observer que malgré une bonne détection de pictogrammes sur certaines imagettes 6(a), il arrive que le modèle détecte des éléments qui ne correspondent pas à des pictogrammes de marais, comme un quarroyage bleu présent sur

certaines cartes anciennes, ou des pictogrammes correspondant à une autre couche que celle des marais 6(b). Ces problèmes peuvent être résolus en optimisant le dataset en prenant en compte ces éléments dans la création de fausses cartes anciennes. Le modèle serait ainsi capable de ne PAS assimiler des objets bleus comme ceux-ci, à des pictogrammes de marais.

En comparant les évaluations de performance des différents modèles ainsi que les images tests qui en ressortent (cf figure 21 en Annexe), nous en avons déduit que les valeurs d'hyper-paramètres optimales seraient une taille de lot de 32 et 128 epochs. Nous aurions potentiellement obtenu de meilleurs résultats avec un nombre plus important d'epochs, avant d'atteindre le sur-apprentissage, mais cela n'a pas pu être vérifié pour cause de manque de temps et de matériel performant.

Ultérieurement, il sera possible pour nos commanditaires d'implémenter une fonctionnalité dans le code pour récupérer la position en coordonnées géographiques des pictogrammes détectés et les stocker dans un fichier GeoJson afin de facilement les ajouter dans une BD topo ou un SI.

5 Réalisation et suivi de projet

5.1 Choix du nom de notre projet

Le nom que nous avons retenu pour notre projet parmi toutes les idées que nous avons pu avoir est **SWAMPY**.

Swamp est un mot anglais pour signifier marais. L'ajout du préfixe Py permet de rendre notre outil plus attachant. Les modèles IA ont souvent des petits noms de ce type, nous avons par exemple, les IA aux échecs comme *DeepBlue*, *StockFish* ou les IA d'entreprise comme Apple avec *Siri* et Google avec *Alexia*. De plus, le préfixe Py fait office notamment de clin d'oeil au langage Python comme notre modèle sera codé en ce langage de programmation. Les pictogrammes que notre outil doit détecter nous font penser à une dorsale de Spinosaurus c'est pourquoi notre logo est composé d'un dinosaure de cette espèce avec un pictogramme pour dorsale. Il y a beaucoup d'outils/logiciels qui utilisent des animaux dans leur logo (un éléphant pour pgAdmin, un dauphin pour mySQL, un pingouin pour Linux etc.), nous n'avons pas encore rencontré encore de logo avec des dinosaures donc c'est assez original.

5.2 Logo



5.3 Organisation

Sur le plan organisationnel, nous avons désigné Suzanne Lizot en tant que cheffe de projet. Tous les membres de l'équipe sont développeurs.

5.4 Planning prévisionnel

	Février	Mars	Avril	Mai								
	7	14	28	6	18	20	27	3	24	29	30	3
Phase 0 : Réflexions sur le sujet												
Recueil du besoin	1/2/3/4	1/2/3/4										
Établissement des étapes du projet	1/2/3/4	1/2/3/4										
Conception du planning prévisionnel		2	2									
Attribution des rôles		1										
Préparation du COPIL de lancement	1/2/3/4	1/2/3/4										
Phase 1 : Création de jeux de données d'apprentissage												
Tests de différentes méthodes de répartition de points			3									
Python				1								
Web					1							
Tests de différentes méthodes d'ajout de pictogrammes					4							
QGIS						3						
Python							1					
Web								1				
Tests de différentes méthodes de génération de masques								3	3	3	3	
QGIS								3	3	3	3	
Python									3	3	3	
Web										3	3	
Tests de différentes méthodes de maillage/découpage de la France												
QGIS												
Python												
Web												
Comparaison des méthodes	1/3/4	1/3/4	1/3/4									
Phase 2 : Prise en main des modèles de Deep Learning												
U-Net												
Compréhension et appropriation des codes	2	2	2	2	2	2	2	2	2	2	2	
Tests avec des datasets (trouvés sur internet) qui se rapprochent de cartes		2	2	2			2	2	2	2	2	
Evaluation des performances						2	2	2	2	2	2	
Segformer												
Compréhension et appropriation des codes								1	1			
Tests avec des datasets (trouvés sur internet) qui se rapprochent de cartes								1	1			
Evaluation des performances									1	1		
Comparaison des modèles									1/2	1/2		
Phase 3 : Entrainement des modèles avec le dataset créé												
Entrainement des modèles										2	1/2/3/4	1/2/3/4
Optimisation en faisant varier les hyper-paramètres										2	1/2/3/4	1/2/3/4
Evaluation des performances des modèles										2	1/2/3/4	1/2/3/4
Phase 4 : Application sur des cartes anciennes												
Evaluation des performances										2	2	2
Phase 5 : Fin du projet												
Rédaction du rapport					1/2/3/4	1/2/3/4	1/2/3/4	1/2/3/4	1/2/3/4	1/2/3/4	1/2/3/4	1/2/3/4

FIGURE 7 – Planning prévisionnel

Les phases 1 et 2 se sont faites en parallèle pour que chaque scénario puisse être étudié par une personne et qu'une personne étudie en même temps les codes des

modèles de Deep Learning. Ainsi, une fois qu'un scénario de génération de data-set a été validé, nous avons pu toutes ensemble enchaîner avec l'entraînement des modèles. Si un scénario est abandonné, il était prévu que la personne assignée à ce scénario accompagne la personne prenant en main les modèles de Deep Learning, en se répartissant par exemple U-Net et SegFormer.

5.5 Distribution des tâches

RÉPARTITION DES TÂCHES

 Suzanne	Exploration de la piste OGIS pour la création d'un data set	Reproduction de cartes anciennes de type TOP 25 n° 3331 OT	Réussir à découper nos cartes anciennes en plusieurs parties sur qgis	Création d'un code permettant de découper des images et passer de la couleur au binaires pour le masque
 Lilia	Prise en main du machine learning	Reproduction de cartes anciennes de type 2714 E	Essai d'entraînement du machine learning avec nos cartes anciennes	
 Vanessa	Exploration de la piste Web pour la création d'un data set	Mise en place de la technique pour créer des masques à partir des cartes anciennes		
 Elisa	Exploration de la piste Python pour la création d'un data set	Reproduction des pictogrammes de marais des années 1997,2006 et 2007	Rédiger en python les commandes réalisé sur OGIS par Lilia et Suzanne	Rédaction des livrables

5.6 Les risques et difficultés rencontrées

Un des premiers risques dans ce projet est la discorde dans le groupe. La discorde au sein du groupe peut avoir des répercussions significatives sur la performance et le bien être des membres. C'est pour cela qu'il est important d'encourager les membres du groupe à exprimer leurs préoccupations et leurs opinions de manière ouverte et respectueuse.

Le deuxième risque que nous avions envisagé était le fait d'avoir des commanditaires absents. Or ils ont toujours répondu à nos nombreux mails.

Un autre risque était les problèmes de santé. Les problèmes de santé liés au stress et à la pression peuvent avoir un impact significatif sur les membres du groupe tels que le burn-out, l'anxiété et dépression, baisse de la motivation et de l'engagement et donc une diminution de la qualité de vie. Vanessa est tombée malade les deux semaines avant la soutenance, ce qui l'a malheureusement empêché de travailler et de suivre correctement les actions réalisées par les autres membres du groupe. C'est donc dans ce raisonnement là que Suzanne a remplacé Vanessa en tant que cheffe de projet afin d'assurer notamment la communication avec les commanditaires.

De plus, il aurait pu nous manquer des matériels performants pour faire du deep learning. Cela engendrait un ralentissement voire l'impossibilité de faire des calculs sur le processus de développement, d'entraînement et de test des modèles. Cela nous

a aussi poussées à limiter le nombre d'epochs pour parvenir à des résultats dans le temps imparti. Une des solutions serait de demander à l'équipe de RSI de l'école s'ils pouvaient nous prêter des ordinateurs puissants.

Nous avons été confrontés à plusieurs défis lors de ce projet, notamment dans le domaine du deep learning, une discipline que nous avions peu exploré au cours de notre formation académique. L'apprentissage des concepts et des techniques nécessaires a demandé davantage de temps que prévu, ce qui a rallongé la phase de recherche. Pour surmonter ces obstacles, nous avons sollicité l'assistance du service IA de l'IGN et consulté les ressources pédagogiques disponibles, notamment les cours de la filière PPMD portant sur ce thème. Cette étape de recherche intensive a pris le pas sur le développement effectif du code, mais elle s'est avérée cruciale pour la réussite du projet.

6 Conclusion

Ce projet d'analyse, mené en collaboration avec l'IGN, visait à résoudre le problème de la disparition partielle des marais sur les cartes topographiques actuelles. En suivant une méthodologie rigoureuse, nous avons étudié le contexte du projet, établi des objectifs clairs et réalisé une série d'étapes précises, de la création de cartes anciennes à l'entraînement de modèles de deep learning pour la détection des pictogrammes de marais.

Malgré quelques défis rencontrés, notamment dans l'apprentissage du machine learning, notre équipe a su surmonter ces obstacles grâce à une recherche intensive et à une organisation efficace. En résultat, notre solution, baptisée SwamPy, offre une approche innovante pour améliorer la précision et l'exhaustivité des cartes topographiques de l'IGN, ouvrant ainsi de nouvelles possibilités pour la préservation et l'utilisation de ces données précieuses.

7 Annexes



FIGURE 8 – Cartes IGN que nous avions à disposition

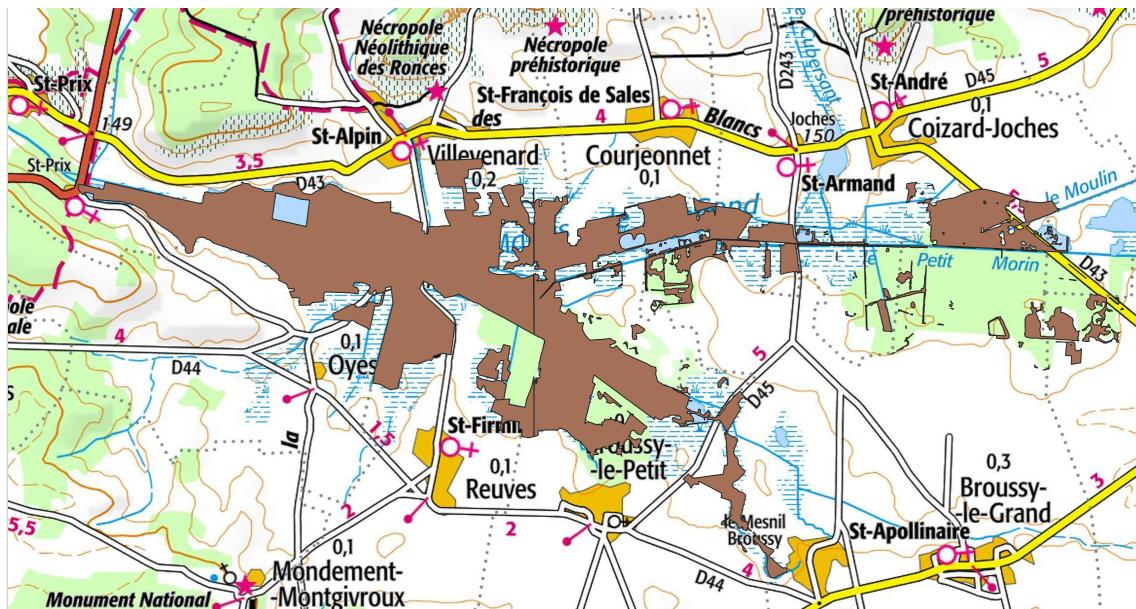


FIGURE 9 – Portion de végétation aquatique représentée en marron sur un flux de carte IGN, capture d'écran sur QGIS



FIGURE 10 – Captures d'écran de marais de cartes anciennes



FIGURE 11 – 3 pictogrammes de marais que nous avons crée selon le style de 1997



FIGURE 12 – 3 pictogrammes de marais que nous avons crée selon le style de 2006



FIGURE 13 – 3 pictogrammes de marais que nous avons crée selon le style de 2007

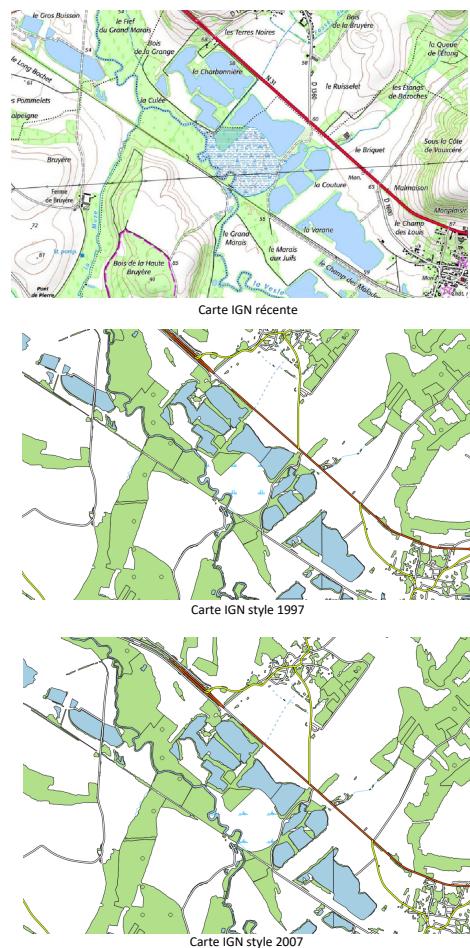


FIGURE 14 – Exemple de l'application de notre style type carte TOP 25 n° 3331 OT sur la même zone d'une carte

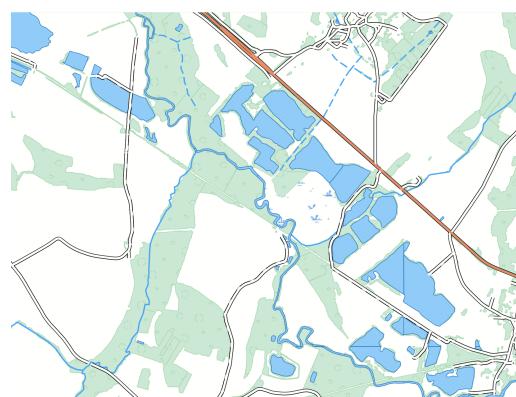


FIGURE 15 – Exemple de l'application de notre style type carte 2714 E sur la même zone que figure 14

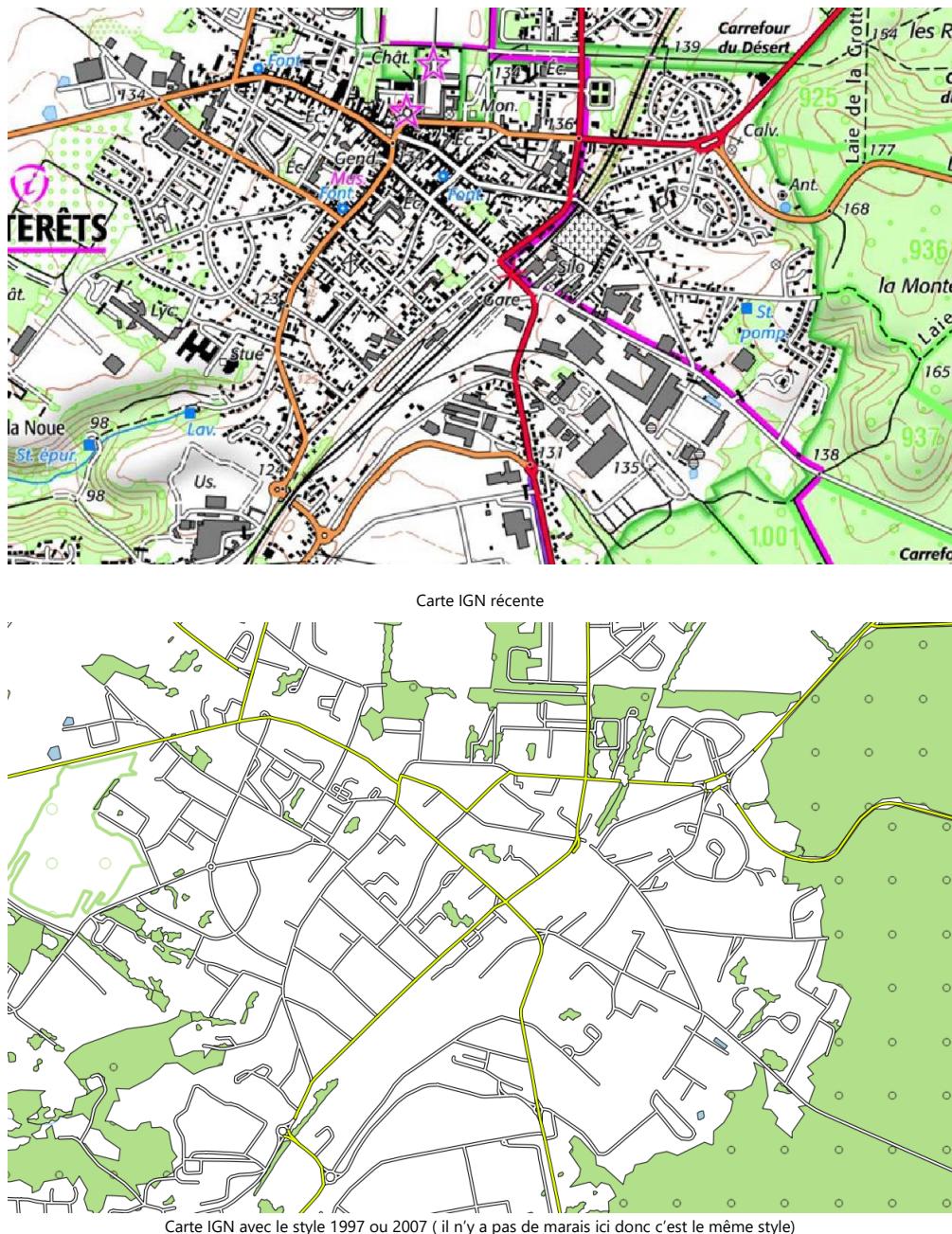


FIGURE 16 – Deuxième exemple de l'application de notre style type carte TOP 25 n° 3331 OT sur la même zone d'une carte

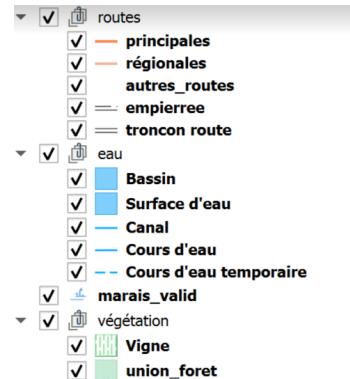
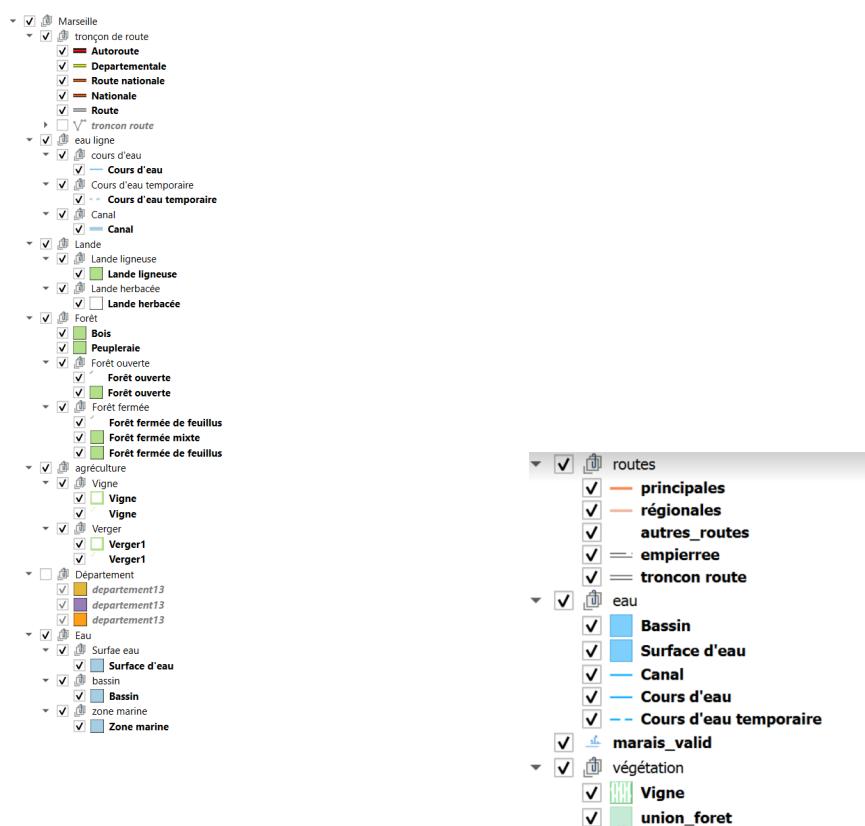


FIGURE 17 – Légendes des couches vectorielles dans nos projets QGIS

Comparaison

Nom de la couche	Scan 25 IGN récent	2006
Forêt fermée de feuillus	vert clair avec des petits et plus grands cercles	
Peupleraie	vert clair avec picto rigolo	
Bois	vert clair	
Forêt fermée mixte	vert clair avec petits et grands cercles et « triangles »	
???	vert clair avec « triangles »	
Forêt ouverte	points verts clairs sur fond blanc	
Lande ligneuse	« v » verts clairs sur fond blanc	
Surface d'eau	bleu clair, contour bleu	
Verger	cercles verts clairs contour vert	
Bassin	bleu clair, contour bleu	
Vigne	petits traits verticaux verts	
Marais		
Régionale	orange	
Autres	blanche	
GR (!= tronçon de route)	blanche et rose	
Principale	rouge	
Route empierrée	contours tirets	
!= tronçon de route	trait noir	

FIGURE 18 – Méthode pour créer le style type carte 2714 E à partir de la BD topo



FIGURE 19 – Imagettes de carte ancienne datant de 1997



FIGURE 20 – Imagettes de carte ancienne datant de 2006

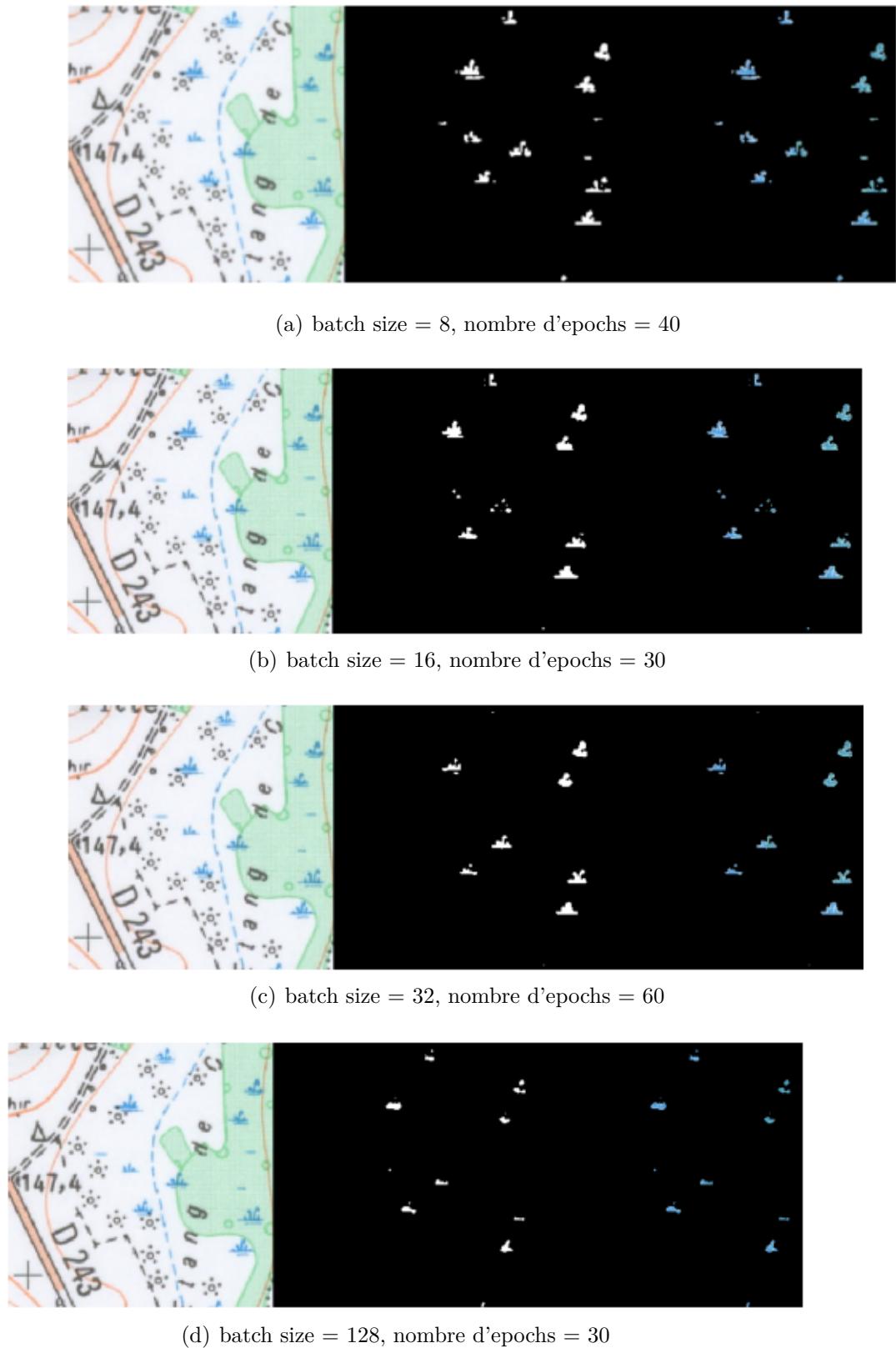


FIGURE 21 – Résultats de tests effectués sur une imagette de carte ancienne datant de 2006