Humanités numériques : un cas pratique avec l'identification de hiéroglyphes

Sarah Madeleine, Sylvain Maillot, Sara Thomas

Humanités numériques : un cas pratique avec l'identification de hiéroglyphes

S. Madeleine[1] S. Maillot[1] S. Thomas[1] M. Servajean[1 2]

[1] Université Paul Valéry Montpellier, Route de Mende, Montpellier 34199, France
{sarah.madeleine, sylvain.maillot, sara.thomas}@etu.univ-montp3.fr

[2] LIRMM, 860 rue de St Priest, Montpellier 34095, France
maximilien.servajean@lirmm.fr

Résumé

Dans le cadre du projet VEgA et des humanités numériques en général, nous nous intéressons à l'utilisation des méthodes de reconnaissance d'image par réseaux de neurones à convolutions et, plus particulièrement, à la classification de hiéroglyphes. Nous présentons ici le modèle de classification utilisé, l'interface graphique permettant son utilisation par le plus grand nombre ainsi que l'analyse des résultats expérimentaux. Bien que disposant de très peu d'images (environ 44 exemples par classe), notre modèle obtient un score supérieur à 90% *en validation.*

Mots Clef

Reconnaissance d'image, apprentissage automatique, réseaux de neurones à convolution, ResNet-18, hiéroglyphes égyptiens, humanités digitales.

Abstract

As part of the VEgA project, we are interested in the use of image recognition methods using convolutional neural networks in the context of digital humanities. We show in particular how classification can be performed on hieroglyphs. We present our classification model and the graphical user interface and we analyse our experimental evaluation. Although we only have few training examples (around 44 *hieroglyphs per class), our model achieves more than* 90% *accuracy on the validation set.*

Keywords

Image recognition, automatic learning, convolutional neural networks, ResNet-18, Egyptian hieroglyphs, digital humanities.

1 Introduction

Ces dernières années ont vu les performances de diverses tâches d'apprentissage statistique, notamment en classification d'images, exploser, et ce, principalement grâce au renouveau de l'apprentissage profond [10]. Cette opportunité s'est initialement cantonnée aux sciences dites dures, où la disponibilité des données nécessaires à l'apprentissage d'un modèle profond est plus immédiate.
De leur côté, les humanités numériques (ou *digital humanities*) [1] se sont tout d'abord manifestées par une révolution des usages avec des plateformes comme LexArt [1], dédiée au lexique artistique de 1600 à 1750, ou VEgA [2] (Figure 1), premier dictionnaire numérique de l'égyptien ancien. L'impact de tels projets se mesure autant dans la centralisation de l'information et que dans leur mise à disposition au travers d'interfaces innovantes.

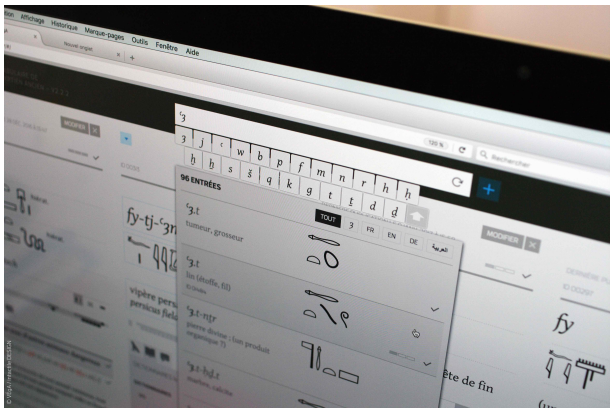


FIGURE 1 – Interface du dictionnaire VEgA.

Cependant, les techniques d'apprentissages statistiques (e.g. classification, régression) ne sont encore que très peu répandues dans ces disciplines qui, pourtant, pourraient en bénéficier [7]. En Egyptologie, objet d'étude de cette contribution, les hiéroglyphes peuvent ainsi être identifiés (i.e. classification) à partir d'images.
La classification consiste, à partir d'une donnée d'entrée $\mathbf{x_i}$, à en prédire la classe (ou le label) y_i. Il s'agit d'une stratégie supervisée dans le sens où nous disposons d'un jeu d'apprentissage $\mathcal{D} = \{(\mathbf{x_1}, y_1), ..., (\mathbf{x_N}, y_N)\}$. Pour ce travail, $\mathbf{x_i}$ représente une image de hiéroglyphe et y_i la détermination (le nom) de ce dernier. Le cas particulier des

1. http ://lexart.fr
2. http ://vega-vocabulaire-egyptien-ancien.fr

images fait face à deux problèmes importants : la malédiction de la dimension et la notion de séparabilité linéaire. La grande dimension favorise le sur-apprentissage [8] et la non séparabilité linéaire [11] empêche tout classifieur linéaire de s'adapter correctement aux données.
Les réseaux de neurones profonds [11] jouent sur ces deux tableaux. Concernant la dimensionnalité, la succession des représentations des données (les couches du réseaux) ferait office de goulot d'étranglement de l'information [16], et l'algorithme d'optimisation stochastique SGD jouerait comme un régularisateur [2] limitant par là même les possibilités de sur-apprentissage. En outre, la succession d'activations non-linéaires permet de "tordre" l'espace d'entrée et de projeter les données dans un espace de dimension réduite où chaque classe devient séparable linéairement des autres. En omettant la dernière couche $g(\mathbf{x})$ de classification linéaire du modèle, l'ensemble du réseau peut ainsi être vu comme permettant l'apprentissage de *features* simples et séparables linéairement des données d'entrées. En d'autres termes, un modèle profond peut donc être vu comme la composée d'une fonction d'apprentissage de *features* f et d'une fonction de classification linéaire g : $(g \circ f)(\mathbf{x})$.
Malheureusement, de tels modèles nécessitent une très grande quantité de données, incomparable avec ce qui est réellement disponible dans le monde des humanités numériques. Ce travail montre qu'il est malgré tout possible, en se concentrant sur des techniques de l'état de l'art, d'obtenir d'excellentes performances (supérieures à 90% de précision en validation). Pour cela, il est par exemple nécessaire de s'appuyer sur la notion de *transfert learning*. En effet, la fonction de coût d'un réseau de neurones admet un grand nombre de minimum locaux. Ainsi, le point de départ de l'apprentissage a un impact sur le point de convergence. Le *transfert learning* s'appuie sur cette idée. En entraînant le modèle sur un jeu de données tiers conséquent, il apprend des *features* visuelles pertinentes et utiles pour la seconde tâche où la disponibilité des données est plus problématique. L'augmentation des données permet également de compenser cette limite en déformant aléatoirement les données d'entrées.
Afin de rendre utilisable notre modèle par le plus grand nombre, nous présentons également l'interface graphique qui l'accompagne. Celle-ci permet à la fois de lancer l'apprentissage d'un modèle lorsque de nouvelles données deviennent disponibles ; et d'identifier des hiéroglyphes par glisser/déposer.
La suite de ce papier se structure comme suit. La section 2 présente les travaux antérieurs. Notre modèle est introduit dans la section 3, et nous montrons ses résultats expérimentaux dans la section 4. Enfin nous concluons ce travail de recherche dans la section 5.

2 Travaux antérieurs

Les articles [4], [7], [9] et [12] font état des travaux de recherche dans le domaine de la reconnaissances des symboles hiéroglyphiques par apprentissage automatique. Dans [9], Kriege et al. ont travaillé sur la reconnaissance de symboles cunéiformes à l'aide de deux méthodes basées sur la mesure de similarité de deux graphes : la méthode du plus proche voisin et celle d'un CNN. La première étant très coûteuse en terme de temps de calcul et nécessitant un volume important de données pour la phase d'apprentissage ; les auteurs se sont tournés vers un CNN. Bien que la démarche d'utiliser un CNN soit identique à la nôtre, leur modèle se base sur un a priori concernant la forme des symboles, ce qui n'est pas notre cas où nous nous appuyons sur les images brutes sans pré-traitement.

Dans [4], Duque-Domingo et al. proposent une méthode de décryptage des cartouches hiéroglyphiques basée sur les techniques d'analyse d'image et vision par ordinateur. Ils identifient notamment les hiéroglyphes présents dans une image grâce à la comparaison des distances de Chamfer et d'Hausdorff d'une image connue (i.e. une image-étalon) à celles de l'image étudiée. Bien que cette méthode s'affranchisse des biais induits par la luminosité de l'image ou par la texture du support du cartouche ; elle nécessite d'avoir à disposition autant d'images-étalons qu'il existe de symboles dans la langue étudiée, en incluant leur diversité. À l'inverse, notre modèle réalise la majorité de son apprentissage sur le jeu de données ImageNet. Cela nous permet de finaliser son apprentissage (fine-tuning) sur un jeu de données, spécifique aux symboles étudiés, restreint et non-exhaustif.

Dans [7] Hu et al. présente la méthode de l'histogramme du contexte de l'orientation de la forme pour identifier les symboles hiéroglyphiques. Cette méthode nécessite un important travail de traitement d'image en amont de l'identification par des épigraphistes pour faire disparaître le bruit. Tandis que notre modèle tire profit du bruit des images pour augmenter ses performances.

De nombreuses architectures ont été proposées et ont obtenu des résultats significatifs en classification d'images [3, 6, 10, 13, 14, 15, 17]. Il ressort que les réseaux de neurones à convolution avec des connections résiduelles sont plus simples à entraîner et obtiennent de meilleurs résultats en terme d'apprentissage que les réseaux sans. Dans [13], les auteurs démontrent notamment que la profondeur du réseau a une influence sur sa précision. De même que dans [17] il est démontré que la méthode dite *"weighted-pooling"* obtient de meilleurs résultats que les méthodes de pooling traditionnelles, telles que *max-pooling, average pooling*.

Nous nous sommes servis de ces travaux pour adapter le modèle ResNet-18 aux contraintes inhérentes à la reconnaissance des hiéroglyphes égyptiens. Nous montrons la faisabilité des modèles CNN à partir du moment où un minimum d'images sont disponibles. Nous montrons les

limites de notre méthode lorsque la variabilité d'un signe est trop grande.

3 Proposition & demonstration

3.1 Jeu de données

Le jeu de données provient de deux sources : des signes du temple de Karnak [3] et d'autres de la pyramide d'Ounas (M. Franken), représentants un total de 4193 symboles répartis en 88 catégories différentes. Cela représente la quantité extrêmement faible de 44 images par classe.
Chaque signe est labellisé à la main selon la liste de Gardiner. Les images de Karnak n'ont pas subi de pré-traitement. Les images du jeu de données de Morris Franken [5] sont recoupées en 50x75. Les pixels vides sont remplis avec une texture rappelant l'arrière-plan réel. Pour augmenter le jeu d'entrainement, les images sont recadrées aléatoirement et transformées par un flip horizontal.

3.2 Architecture

Implémentation PyTorch de ResNet-18 en utilisant des poids pré-entrainés sur le *dataset* ImageNet. Le modèle est constitué de 20 couches : 1 convolution 7x7, 16 convolution 3x3 et 3 couches connectées. La dernière couche donne en sortie une des 88 classes de signes possibles.

3.3 Optimisation

Plusieurs tests d'optimiseur ont été réalisés. Notre choix s'est porté sur Adam avec un *learning rate* initialisé à 0.0001 et divisé par 10 toutes les 7 epochs.

3.4 Activation

Chaque couche contient une opération de normalisation et d'activation. Cette dernière (*ReLU*) introduit une non-linéarité permettant de répondre plus efficacement à notre tâche de reconnaissance et classification d'image. La normalisation (*Batch Normalization*) augmente la stabilité et la vitesse d'apprentissage du modèle en normalisant les sorties des couches cachées à chaque itération.
Le modèle utilise la technique de *DropOut*, initalisée à 0.5, qui consiste en la désactivation temporaire et aléatoire de neurones pendant l'entraînement, puis en leur réactivation lors de la phase de test. Cela simule un apprentissage sur un ensemble de modèles différents. L'objectif est de prévenir les liens d'interdépendance qui pourraient se créer entre les neurones et ainsi éviter l'*overfitting*.

3.5 Entraînement

Le modèle est entrainé sur GPU (GTX 1080 ti) sur 100 epochs.
La fonction d'entropie croisée est utilisée pour évaluer la précision du modèle. La convergence est rapide et dépasse les 90% de précision.

3. http ://sith.huma-num.fr/karnak

3.6 IHM

Afin de rendre possible l'utilisation de notre travail par le plus grand nombre, nous avons développé une interface graphique à l'aide de la librairie Python PYQT5.

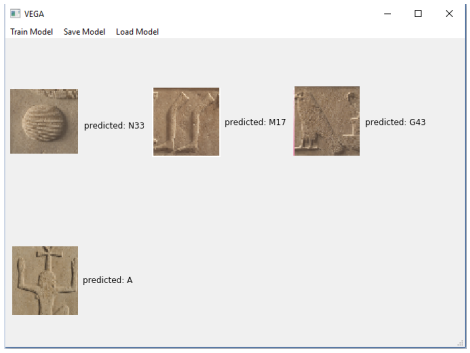


FIGURE 2 – Maquette de l'interface graphique.

Elle est composée d'un bandeau de navigation et d'une zone centrale. Trois onglets compose la barre de navigation : *Train model, Save model* et *Load model*. Le premier onglet permet de calculer un nouveau modèle (e.g. quand de nouvelles images sont disponibles, et viennent enrichir le jeu de données d'apprentissage). A partir du second onglet l'utilisateur peut sauvegarder le modèle pour une réutilisation future. Enfin le chargement d'un modèle déjà existant se fait par le troisième et dernier onglet.
La zone centrale quant à elle, initialement vide, permet d'utiliser la classification. Elle s'utilise en y glissant des images contenant des hiéroglyphes inconnus pour ensuite afficher le résultat.
L'application permet de charger un modèle n'ayant pas nécessairement de lien avec les hiéroglyphes.

4 Évaluation expérimentale

En nous appuyant sur l'état de la recherche dans le domaine de la reconnaissance d'image par apprentissage automatique et des résultats de nos tests, nous sommes parti d'un modèle ResNet-18. Nous avons ensuite choisi les éléments qui correspondaient au mieux à nos besoins pour composer notre modèle final :

- Couches de convolution : 1 taille 7x7, 16 taille 3x3 ;
- Couche de pooling : Max pool et Average pool ;
- Couche de correction : Batchnorm, ReLU ;
- Couche de perte : Cross entropy ;
- Couche connectée : linéaire ;
- Optimiseur : Adam.

Notre modèle est entraîné pendant 100 epochs. Il converge rapidement (i.e. vingtième epoch) vers une valeur limite supérieure à 90% de précision sur le jeu de validation.
La précision de notre modèle est meilleure en phase de validation que lors de celle d'apprentissage. Nous expliquons cela par les transformations infligées au jeu

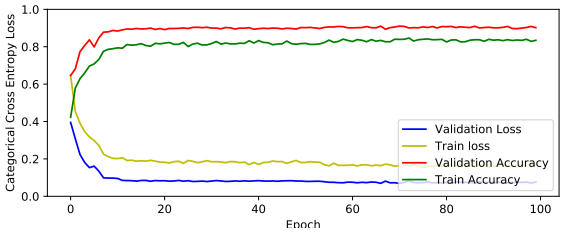
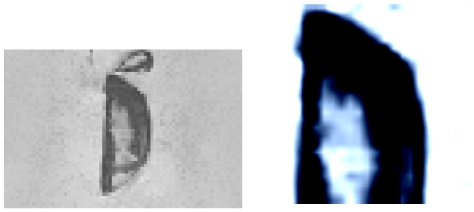


FIGURE 3 – Performances du modèle.

de données d'apprentissage, relativement importantes et complexifiant ainsi la tâche de classification. À l'inverse, le jeu de validation ne subit aucune transformation.



(a) Sans transformation. (b) Avec transformation.

FIGURE 4 – Image d'un signe V31.

5 Conclusion

Notre travail de recherche évalue les possibilités d'utilisation de modèles profonds pour la reconnaissance et la classification automatiquement des hiéroglyphes. Le déploiement d'un modèle résiduel simple à 20 couches apporte des résultats concluants. Il est cependant important de noter que l'efficacité des modèles profonds est largement impactée par la volumétrie et la variété (que l'on retrouverait en test) du jeu de données d'apprentissage.
Notre modèle converge très rapidement. Un modèle plus profond apporterait de meilleurs résultats de classification mais demanderait plus de temps d'entrainement. L'IHM est simple et pourrait à l'avenir proposer d'autres fonctionnalités.

Références

[1] David M Berry and Anders Fagerjord. *Digital humanities : knowledge and critique in a digital age*. John Wiley & Sons, 2017.

[2] Pratik Chaudhari and Stefano Soatto. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. *CoRR*, abs/1710.11029, 2017.

[3] François Chollet. Xception : Deep learning with depthwise separable convolutions. *arXiv preprint*, 2016.

[4] Jaime Duque-Domingo, Pedro Javier Herrera, Enrique Valero, and Carlos Cerrada. Deciphering egyptian hieroglyphs : towards a new strategy for navigation in museums. *Sensors*, 17(3) :589, 2017.

[5] Morris Franken and Jan C. van Gemert. Automatic egyptian hieroglyph recognition by retrieving images as texts. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 765–768, New York, NY, USA, 2013. ACM.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[7] Rui Hu, Carlos Pallán Gayol, Jean-Marc Odobez, and Daniel Gatica-Perez. Analyzing and visualizing ancient maya hieroglyphics using shape : From computer vision to digital humanities. *Digital Scholarship in the Humanities*, 32(suppl_2) :ii179–ii194, 2017.

[8] Eamonn Keogh and Abdullah Mueen. Curse of dimensionality. In *Encyclopedia of Machine Learning and Data Mining*, pages 314–315. Springer, 2017.

[9] Nils M Kriege, Matthias Fey, Denis Fisseler, Petra Mutzel, and Frank Weichert. Recognizing cuneiform signs using graph based methods. *arXiv preprint arXiv :1802.05908*, 2018.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[11] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553) :436, 2015.

[12] Edgar Roman-Rangel and Stephane Marchand-Maillet. Assessing deep learning architectures for visualizing maya hieroglyphs. In *Mexican Conference on Pattern Recognition*, pages 137–146. Springer, 2017.

[13] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv :1409.1556*, 2014.

[14] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.

[15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. Cvpr, 2015.

[16] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. *CoRR*, abs/1503.02406, 2015.

[17] Xiaoning Zhu, Qingyue Meng, Bojian Ding, Lize Gu, and Yixian Yang. Weighted pooling for image recognition of deep convolutional neural networks. *Cluster Computing*, pages 1–13, 2018.