MaxiMask: Identifying contaminants in astronomical images using convolutional neural networks

Maxime Paillassa¹, Emmanuel Bertin², Hervé Bouy¹

¹Laboratoire d’astrophysique de Bordeaux, Univ. Bordeaux, CNRS, B18N, allée Geoffrey Saint-Hilaire, 33615 Pessac, France
²Sorbonne Université, CNRS, UMR 7095, Institut d’Astrophysique de Paris, 98 bis bd Arago, 75014 Paris, France
1. Introduction

2. Convolutional neural networks (CNNs)

3. MaxiMask dataset

4. MaxiMask model and results

5. Conclusion
Introduction
Contaminants in astronomical images

- Electronic
  - Hot/Bad pixels
  - Saturated pixels
  - Persistence effects

- Optic
  - Fringes patterns
  - Diffractions spikes
  - Reflection/Refractions

- External events
  - Cosmic rays
  - Satellite trails
  - Nebulosities
MaxiMask aims

- Astronomical image analyses are largely complicated by contaminants
  → Design a contaminant identifier
- Will to have a robust and performant *All-in-One* tool:
  - Not heuristic based
  - Without a lot of tunable parameters
  - Usable in wide ranges of conditions
  → Use machine learning and convolutional neural networks
Convolutional neural networks (CNNs)
Before talking about CNNs, just NNs and supervised learning:

**Figure 1:** Principles of supervised learning with neural networks. Learning consists of iterating learning steps over the data set.
Convolutional Neural Networks (1/3)

- Take advantage of the convolution operation:
  - Efficient processing of grid like data, even with high dimensions
  - Use a small number of trainable parameters (convolution kernels)
  - Capture translation invariant data features

**Figure 2**: Left: convolution applied to a given pixel. Right: convolution applied to a whole image
Figure 3: Typical CNN architecture for image classification (LeCun et al. 1995)
Convolutional Neural Networks (3/3)

- CNNs automatically learn from raw data the relevant data representation to solve a task

**Figure 4:** Feature map visualization across layers of a handwritten digit recognizer CNN (Harley 2015)
CNNs for semantic segmentation

**Figure 5:** Typical CNN architecture for semantic segmentation (Badrinarayanan et al. 2015)
MaxiMask dataset
Building the input/output dataset

- Identify the cleanest images of our data (Cosmic Dance survey (Bouy et al. 2013))
- Extract and \textit{clean} images from it

\textbf{Figure 6:} Left: extracted image. Right: cleaned image (CTIO-DECam)
Figure 7: Left: extracted image. Right: cleaned image (CFHT-MegaCam)
Figure 8: Left: extracted image. Right: cleaned image (HSC)

• Then add contaminants in these clean images
Start from a clean image
Add contaminants (1/8)
Add contaminants (2/8)
Add contaminants (3/8)
Add contaminants (4/8)
Add contaminants (6/8)
Add contaminants (7/8)
Add contaminants (8/8)
Inherent-to-data contaminants
Learning data samples

Figure 9: Left: input image. Right: ground truth.
MaxiMask model and results
MaxiMask CNN model

Figure 10: MaxiMask CNN model architecture (Yang et al. 2018)
MaxiMask training

- 50,000 images $400 \times 400$
- 30 epochs, Adam optimizer (Kingma et al. 2014)
- Tensorflow (Abadi et al. 2016)
- Nvidia TITAN X GPU

→ Approximately 24 hours

**Figure 11**: 24 of the 32 first layer feature maps
Figure 12: Left: input image. Center: ground truth. Right: prediction.
Quantitative test set results (1/2)

- True Positive Rate = $TPR = \frac{TP}{P} = \frac{TP}{TP+FN}$
- False Positive Rate = $FPR = \frac{FP}{N} = \frac{FP}{TN+FP}$
- Purity or Precision = $PUR = \frac{TP}{TP+FP}$

Figure 13: Left: TPR vs FPR. Right: TPR vs PUR.
Quantitative test set results (2/2)

Bad columns/lines/clusters ROC curve
AUC: 0.99881

Persistence ROC curve
AUC: 0.98216

Satellite ROC curve
AUC: 0.99567

Fringes ROC curve
AUC: 0.97870
Real life result examples

Figure 14: Left: A real image from an HSC CCD. Center: recovering cosmic rays. Right: recovering satellite trail
Conclusion
Conclusion

- We can identify contaminants in images using CNNs
- MaxiMask available for inference at: https://github.com/mpaillassa/MaxiMask
- Still some contaminants to include/improve:
  - diffraction spikes
  - ghosts
  - reflections
  - infrared detectors contaminants
- Current work in progress to apply it to spatial data as well (Euclid)
Thank you!
Questions?
**Tensorflow: A system for large-scale machine learning.**  

V. Badrinarayanan, A. Kendall, and R. Cipolla.  
**Segnet: A deep convolutional encoder-decoder architecture for image segmentation.**  

E. Bertin and S. Arnouts.  
**Sextractor: Software for source extraction.**  
**Dynamical analysis of nearby clusters. Automated astrometry from the ground: precision proper motions over a wide field.**  

A. W. Harley.  
**An interactive node-link visualization of convolutional neural networks.**  

D. P. Kingma and J. Ba.  
**Adam: A method for stochastic optimization.**  
Y. LeCun, Y. Bengio, et al.
Convolutional networks for images, speech, and time series.

M. D. Richard and R. P. Lippmann.
Neural network classifiers estimate bayesian a posteriori probabilities.

T. Yang, Y. Wu, J. Zhao, and L. Guan.
Semantic segmentation via highly fused convolutional network with multiple soft cost functions.
MaxiMask cost function

- **Final predictions loss:**
  - \[ L_f = -\frac{1}{B} \sum_{b} \sum_{p,c} (y_{b,p,c} \log(\hat{y}_{b,p,c}) + (1 - y_{b,p,c}) \log(1 - \hat{y}_{b,p,c})) + L_{2_{\text{reg}}} \]
  - \( B = \) Batch Size; \( P = \) Set of all batch pixels; \( C = \) Set of all classes.
  - \( \hat{y}_{b,p,c} = \) Sigmoid class \( c \) prediction of pixel \( p \)
  - \( y_{b,p,c} = \) Class \( c \) ground truth of pixel \( p \), i.e:
    \[
    y_{b,p,c} = \begin{cases} 
      1 & \text{if } p \in c \\
      0 & \text{otherwise}
    \end{cases}
    \]
  - Each pixel is weighted according to its class (and its 8-neighbors):
    \[
    w_c = \frac{1}{p_c \sum_{c'} \frac{1}{p_{c'}}}
    \]
  - The total loss is a combination of the final loss and the pre-prediction losses
Bayesian prior modification

- Output probabilities can be interpreted as Bayesian posteriors (Richard et al. 1991)
- Priors (= Class proportions) can be modified to adapt the output probabilities to new (expected) class proportions:

\[
P(c|x) = \frac{P_N(c|x)}{P_N(c|x) + \frac{P_N(c)}{1-P_N(c)} \frac{1-P(c)}{P(c)} (1 - P_N(c|x))}
\]

- \(P(c|x)\) = New posteriors
- \(P_N(c|x)\) = Raw neural network posteriors
- \(P(c)\) = Class \(c\) new prior (expected class \(c\) proportion in data)
- \(P_N(c)\) = Class \(c\) training prior (class \(c\) proportion in training data)