ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

Parallelized rotation and flipping INvariant Kohonen maps (*PINK*) on *GPUs*

Polsterer K.L.[1], Gieseke F.[2], Igel C.[3], Doser B.[1], and Gianniotis N.[1] *

1- HITS gGmbH (Heidelberg Institute for Theoretical Studies) - Astroinformatics
Schloss-Wolfsbrunnenweg 35, 69118 Heidelberg - Germany

2- Radboud University Nijmegen - Institute for Computing and Information Sciences
Toernooiveld 212, 6525 AJ Nijmegen - The Netherlands

3- University of Copenhagen - Department of Computer Science
Sigurdsgade 41, 2200 København N - Denmark

Abstract. Morphological classification is one of the most demanding challenges in astronomy. With the advent of all-sky surveys, an enormous amount of imaging data is publicly available. These data are typically analyzed by experts or encouraged amateur volunteers. For upcoming surveys with billions of objects, however, such an approach is not feasible anymore. In this work, we present a simple yet effective variant of a rotation-invariant self-organizing map that is suitable for many analysis tasks in astronomy. We show how to reduce the computational complexity via modern *GPUs* and apply the resulting framework to galaxy data for morphological analysis.

1 Introduction

For modern astronomical surveys like the *Sloan Digital Sky Survey* (*SDSS*) [1], volunteers have manually performed various tasks quite successfully (e.g., classification). Within the *Galaxy Zoo* project [2], about 100k volunteers performed a morphological analysis of roughly one million galaxies. However, the projected increase in the number of objects for the next generation of all-sky survey missions renders such a manual inspection impossible. Furthermore, the labelling process may introduce biases, as the volunteers often lack the required scientific background. As crowd-sourcing project are designed for specific tasks, new scientific questions will require new crowd-sourcing projects. Therefore novel explorative analysis methods are required.We promote a semi-automatic data analysis scheme that combines unsupervised learning with the visual recognition capabilities, the creativity, and keen perception of the human brain.[1]

Computers are ideal in pre-processing and pre-analyzing data. We aim for assisting astronomers by making the fully manual analysis via crowd-sourcing projects obsolete. By combining similar and frequent objects via machine learning models, we can retrieve single representatives and thus reduce the number

*KLP,BD,NG gratefully acknowledge the support of the Klaus Tschira Foundation. CI gratefully acknowledges support from The Danish Council for Independent Research through the project "Surveying the sky using machine learning".

[1]A preliminary version of this work not containing the *GPU* implementation and restricted to single-channel image data has been presented [3].

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

of objects that require a manual inspection. The goal of this work is to enable astronomers to efficiently perform a morphological analysis on huge amounts of pre-analyzed data (e.g., images or radio-synthesis data).

In astronomy, morphological features are usually extracted and a simple model is used for classification [4]. Recently, a supervised classification of galaxy shapes based on convolutional neural networks has been proposed [5]. Even though the presented approach is very creative and novel to an application in astronomy, it suffers from the strong biases given in the training data. Unsupervised methods could help to understand the data and to create a better reference sample for supervised processing. In the past, dimensionality reduction techniques that compute topological maps, i.e., latent embeddings, have shown good results [6]. Those techniques aim at projecting complex, high-dimensional data to a low-dimensional presentation while preserving similarities and neighborhood relations between the original data points. We present a rotation and flipping invariant similarity measure and resort to *self-organizing maps* (*SOM*) [7] to obtain a visual representation of the data. Since the used similarity measure is computationally very expensive, we make use of an efficient parallel implementation that can take advantage of both, modern multi-core *CPU* systems as well as massively-parallel *GPU*-based environments.

2 Method

We present the *Parallelized rotation/flipping INvariant Kohonen map* (*PINK*[3]) framework that generates a compact visualization in an unsupervised way and, thus, permits a semi-automatic analysis of the data by an expert. *SOMs* are a simple yet effective dimensionality reduction technique, which are a specialized form of neural networks where every fixed node/neuron $\mathbf{p} \in P$ of the latent space contains a derived prototype after having trained the model.

2.1 Kohonen Maps

As *SOMs* are easy to implement it was possible to develop an implementation for our use-case that runs very efficiently in a parallel environment. We provide a quick overview of *SOMs* in order to understand the efforts undertaken to speed-up the calculations. The neurons $P = \{\mathbf{p}_j = (\mathbf{w}_j, \mathbf{c}_j) \mid \mathbf{w}_j \in \mathbb{R}^d, \mathbf{c}_j \in \mathbb{N}^2, j = 1, \ldots, \mu_P\}$ map every prototype or weight-vector $\mathbf{w}_j$ to a coordinate $\mathbf{c}_j$ in the map. Therefore the trained maps provide both, a data compression via prototypes as well as a spatial ordering based on similarity. In the training phase, the n patterns $\mathbf{y}_i \in \mathbb{R}^d$ with $i = 1, \ldots, n$ are iteratively applied to the map. By calculating a similarity measure $\Delta(\mathbf{y}, \mathbf{w}_j)$ between a pattern $\mathbf{y} \in \mathbb{R}^d$ and the weight $\mathbf{w}_j \in \mathbb{R}^d$ for every node $\mathbf{p}_j \in P$ of the map, the closest (winning) neuron $q(\mathbf{y}) = \text{argmin}_{j=1,\ldots,\mu_P}\ \Delta(\mathbf{y}, \mathbf{w}_j)$ is determined. In our case $\Delta(\mathbf{y}, \mathbf{w}_j)$ is calculated in a rotation invariant way (see Section 2.2). Then the neurons are updated based on the distance to the winning neuron in the map $d(\mathbf{c}_{q(\mathbf{y})}, \mathbf{c}_j)$ and the number of applied iterations t via a training function $f(d(\mathbf{c}_{q(\mathbf{y})}, \mathbf{c}_j), t)$. This is done by updating the weight-vector $\mathbf{w}_j$ of a neuron $\mathbf{p}_j$ to the new value

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

$\mathbf{w}'_j = \mathbf{w}_j + (\psi^{(j)}(\mathbf{y}) - \mathbf{w}_j) \cdot f(d(\mathbf{c}_{q(\mathbf{y})}, \mathbf{c}_j), t)$, where $\psi^{(j)}$ is the identity function in the standard Kohonen-map algorithm and will be used here to align the coordinate systems of $\mathbf{y}$ and $\mathbf{w}_j$. From this formal description it is obvious, that the ordering, the amount of objects of a certain class, the chosen training function, and the number of iterations have a significant impact on the result. This is a weakness of *SOMs* one always should keep in mind when inspecting the results.

2.2 Similarity Measure

As described above, the training of the *SOM* depends on the similarity measure $\Delta(\mathbf{y}, \mathbf{w}_j)$ between the image $\mathbf{y}$ and the weights $\mathbf{w}_j$ of the neuron $\mathbf{p}_j$. Humans can easily scale, align, distort, and interpolate images and are very powerful in checking similarities. Pre-processing the images to align them to the principal axis of their main component and using a simple pixel-wise Euclidean distance was one of the first approaches to deal with rotation. In the past, we carried out multiple tests with rotation invariant similarity measures with the best results achieved via Fourier transformed circular slices of the images [8]. For the imaging data at hand, a rotation and flipping invariant similarity measure is essential to achieve satisfying results. All images had been centered and scaled in a pre-processing step to make them invariant against this kind of deviation.

To calculate the similarity, we basically calculate the Euclidean distances for all possible rotations/flipped objects in the map to determine the best match. It can be shown that this operation still gives rise to a valid distance metric: Let $\Delta(\mathbf{A}, \mathbf{B}) = \min\{d(\mathbf{A}, \phi(\mathbf{B})) \,|\, \phi \in \Phi\}$ and $\Phi = \{\phi_1, \ldots, \phi_N\}$ be a set of image transformations $\phi_i : X \mapsto X$ in the space of the images X. We assume that [I] d is a metric on X, [II] Φ forms an Abelian group, and [III] for any $\phi \in \Phi$, $d(\phi(\mathbf{A}), \phi(\mathbf{B})) = d(\mathbf{A}, \mathbf{B})$. The idea of the proof is that Δ is a metric on $X/\!\sim$, where $\mathbf{A} \sim \mathbf{B}$ if $\mathbf{A} = \phi(\mathbf{B})$ for some $\phi \in \Phi$. Identity is obvious while symmetry is based on the symmetry of d and assumption [III]. Triangular inequality is shown via $\Delta(\mathbf{A}, \mathbf{C}) + \Delta(\mathbf{C}, \mathbf{B}) = d(\mathbf{A}, \phi_a(\mathbf{C})) + d(\mathbf{C}, \phi_b(\mathbf{B})) \leq d(\mathbf{A}, \phi_a \circ \phi_b(\mathbf{B})) = \Delta(\mathbf{A}, \mathbf{B})$.

2.3 Implementation

Our *PINK* implementation supports the use of a Gaussian or a Mexican hat as the distance component and makes use of a simple linear damping based on the number of iterations t. Other distance components as well as iteration-dependent functions can be easily added. The framework allows to train quadratic as well as hexagonal maps, both in a continuous repeating or edge-limited version. The hexagonal shape has six instead of just four distinct directions and just allows for integer distances. With the quadratic alignment, even three dimensional cube-shaped projections can be generated. In addition to the map shape, the ability to deal with multi-band image cubes was implemented. After the training phase is finished, the images $\mathbf{y}$ are matched to the derived prototypes P. By inspecting and annotating those prototypes, a scientist is able to classify all

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

matching objects at once. Therefore the amount of objects to be inspected is reduced to the number of prototypes in the map.

2.4 Speed-up via Parallelization

Since the considered brute-force comparisons between an image $\mathbf{y}$ and all the neurons P are computationally very demanding, this task is an ideal candidate for massively parallel implementations. The training algorithm consists of three parts: The generation of the rotated and flipped images, the calculation of the Euclidean distance between every rotated image and neurons, and the update of the neurons.

Iteratively, all rotated and flipped versions of the image $\mathbf{y}$ are created in parallel by using a set of image transformations $\Phi = \{\phi_1, \ldots, \phi_N\}$. To avoid artifacts the rotated images are cropped. The rotation and cropping steps are combined to reduce the size of the region that has to be calculated. The rotations between $0°$ and $90°$ were calculated explicitly, while the remaining ones as well as the flipped versions are generated through simple matrix operations.

Afterwards, the Euclidean distance between each pre-processed image and the neurons is calculated in parallel. This calculation is the most time-consuming part while finding the minimum is one single scan and therefore fast. For an optimal memory access those calculations are split in two steps, as each thread needs access to just one rotated image. The first step is calculating the summed Euclidean distance between every neuron and image hierarchically, whereas the second step is used to determine the minimum value. To speed up the calculation, a parallel reduction scheme is used that ensures a very efficient accumulation [9] on the used hardware (*NVIDIA Tesla K20m*) with a blocksize of 256 as optimal local storage size. The usage of multiple *GPUs* is also implemented resulting in a poor speedup of about 1.3 using two *Tesla K20m* because of the large memory transfer from one device to the other.

The update part of the weights $\mathbf{w}_j$ is a straightforward implementation performed in parallel. Here, the alignment of the training image $\mathbf{y}$ with respect to the neurons has to be considered via a proper choice of ϕ. To avoid unnecessary *if* statements for the distance and distribution function within the *CUDA* kernel, functors are used as template arguments for a static binding.

3 Experimental Evaluation

To evaluate the performance and usability of our approach, we performed experiments on both, synthetical as well as on astronomical data.

3.1 Data

As the synthetical data was just used to ensure the reproducibility of simple geometric shapes, only the results on the real-world data are shown. In our experiment we used radio-synthesis data taken from the *Radio Galaxy Zoo* project.[2]

[2] `http://radio.galaxyzoo.org`

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.

Description	$\#CPUs$	$\#GPUs$	Time	Speed-up
Python Prototype	8	—	17 days	—
C Code, Sandy-Bridge	12	—	$4^h42^m40^s$	87
CUDA Code, *Tesla K20m*	1	2,496	40^m22^s	606
block optimized *CUDA* Code	1	2,496	35^m22^s	689
block optimized *CUDA* Code	1	$2 \times 2,496$	27^m40^s	885

Table 1: Performance achieved with different versions of the code on different hardware when training on 200 k images with 2° rotation steps for one iteration. As a reference for the calculation of the speed-up factor the Python prototype from the original publication [3] is used.

Regular cutouts with 128 px × 128 px were used to create rotated versions of 86 px × 86 px without having non-valid pixels in the corners. In addition, the intensities of the images were normalized to $[0, 1]$ and every pixel value below a 2σ level w.r.t. the background noise was masked as background and set to zero. Finally, a hexagonal map was trained with the pre-processed data.

3.2 Results

The duration of the training with the 200 k images from *Radio Galaxy Zoo* was used for benchmarking. In Table 1 a comparison between different implementations is given, indicating that the initial calculation time based on a parallel Python code could be enormously improved. The map was trained for ten it-

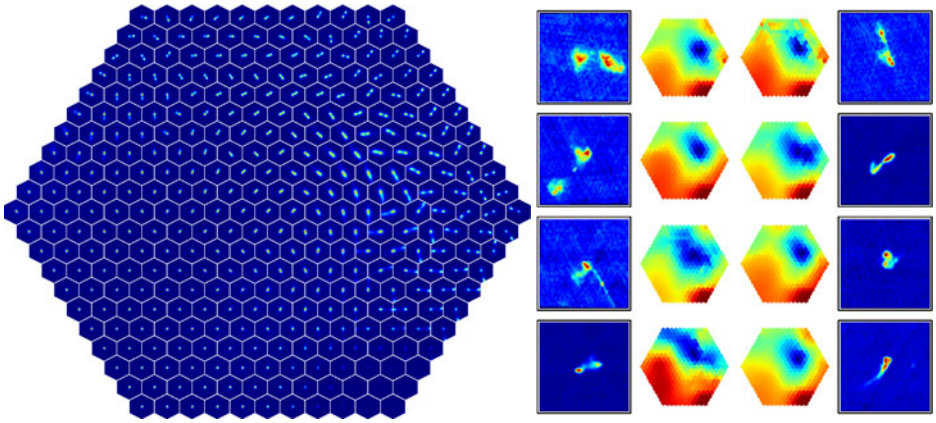

Fig. 1: **left:** Resulting hexagonal Kohonen-map containing the derived prototypes. **right:** Some outlier examples which have been selected based on the quality of their fit to the prototypes in the map (*blue* denotes similar, *red* dissimilar nodes). The corresponding heatmaps indicate potential prototypes they could belong to, even though they are not as well represented as the majority of objects.

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

erations. Already after four iterations no obvious changes had been observed when comparing the regular snapshots after $10,000$ images with each other. In the end we retrieved the map presented in Figure 1 *left*. The prototypes of the resulting map allow a clear separation into different morphological classes. Based on the mapping induced by the prototypes, it is possible to transfer the annotations created for the map directly to every individual image. Objects that are not well represented by the prototypes can be filtered out based on the absolute similarity value of the best match. They can be considered as interesting objects that might require additional manual inspection by an expert. In Figure 1 *right*, some of these automatically determined outliers are shown. A manual inspection of all extracted outliers reveals solely objects exhibiting interesting morphological features.

4 Conclusion

The proposed method shows that unsupervised dimension reduction techniques can help astronomers to analyze huge amounts of data. Besides retrieving a classification scheme, one is able to efficiently detect outliers. Those are the objects which need to be analyzed by an expert. The majority of similar objects just needs to be inspected on the basis of a few representatives. This enables astronomers to deal with the upcoming large imaging surveys.

References

[1] C. P. Ahn, R. Alexandroff, C. Allende Prieto, F. Anders, S. F. Anderson, T. Anderton, B. H. Andrews, É. Aubourg, S. Bailey, F. A. Bastien, and et al. The Tenth Data Release of the Sloan Digital Sky Survey: First Spectroscopic Data from the SDSS-III Apache Point Observatory Galactic Evolution Experiment. *ApJS*, 211:17, 2014.

[2] C. Lintott, K. Schawinski, S. Bamford, A. Slosar, K. Land, D. Thomas, E. Edmondson, K. Masters, R. C. Nichol, M. J. Raddick, A. Szalay, D. Andreescu, P. Murray, and J. Vandenberg. Galaxy Zoo 1: data release of morphological classifications for nearly 900 000 galaxies. *MNRAS*, 410:166–178, 2011.

[3] K. L. Polsterer, F. Gieseke, and C. Igel. Automatic Classification of Galaxies via Machine Learning Techniques: Parallelized Rotation/Flipping INvariant Kohonen Maps (PINK). In A. R. Taylor and E. Rosolowsky, editors, *Astronomical Data Analysis Software and Systems XXIV*, volume 495, pages 81–86, 2015.

[4] D. B. Wijesinghe, A. M. Hopkins, B. C. Kelly, N. Welikala, and A. J. Connolly. Morphological classification of galaxies and its relation to physical properties. *MNRAS*, 404:2077–2086, 2010.

[5] S. Dieleman, K. W. Willett, and J. Dambre. Rotation-invariant convolutional neural networks for galaxy morphology prediction. *MNRAS*, 450:1441–1459, 2015.

[6] O. Kramer, F. Gieseke, and K. L. Polsterer. Learning morphological maps of galaxies with unsupervised regression. *Expert Systems with Applications*, 40(8):2841 – 2844, 2013.

[7] T. Kohonen. *Self-Organization and Associative Memory*. Springer, 1989.

[8] K. L. Polsterer, F. Gieseke, and O. Kramer. Galaxy Classification without Feature Extraction. In P. Ballester, D. Egret, and N. P. F. Lorente, editors, *Astronomical Data Analysis Software and Systems XXI*, volume 461, page 561, 2012.

[9] M. Harris. Optimizing parallel reduction in CUDA, 2007. NVIDIA Developer Technology.