Universiteit Leiden
The Netherlands

Opleiding Informatica

A Quantum-Inspired Algorithm
for Galaxy Classification

Max Veraar

Supervisor:
Evert van Nieuwenburg

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

25/06/2024

Abstract

The advancements in astronomical data collection have necessitated more efficient and accurate methods for galaxy classification. This thesis explores the development and application of a quantum-inspired algorithm to enhance the classification accuracy of galaxies. Using techniques from machine learning and computer vision, the study employs convolutional neural networks (CNNs) and integrates singular value decomposition (SVD) and tensor networks to process and analyze astronomical images. The proposed method aims to reduce computational complexity while maintaining high classification performance. The results demonstrate that the quantum-inspired algorithm achieves a classification accuracy that, compared to traditional methods, is almost as good, but in most instances they use less data. But all methods significantly perform better than the baseline proposed in this thesis. Additionally, this research highlights the potential of quantum computing concepts in addressing complex problems in astronomy, paving the way for further exploration and application in various scientific domains.

Acknowledgements

I would like to thank my supervisor Dr. Evert van Nieuwenburg for providing guidance and supervision for this thesis. Furthermore, I would like to thank Dr. Adrián Pérez Salinas for helping with solving difficult problems during this thesis. Lastly, I'd like to thank my family for their support.

Contents

1 Introduction

The classification of galaxies has always been a fundamental challenge in the field of astronomy [1]. The exponential growth in the volume of astronomical data, driven by advances in telescope technology and data acquisition methods, for example the launch of the James Webb Space Telescope [2] in late 2021 and the release of the telescope's first of many images in July 2022, cause traditional manual classification techniques to become impractical due to time consumption. Automated and accurate galaxy classification methods are now essential for managing and interpreting the large datasets generated by modern astronomical surveys.

Machine learning and computer vision have emerged as powerful tools for tackling classification problems across various domains, including astronomy. Convolutional neural networks (CNNs), in particular, have shown remarkable success in image recognition tasks. However, the high computational cost and complexity associated with these methods necessitate the exploration of more efficient algorithms.

This thesis investigates the application of a quantum-inspired algorithm for galaxy classification. By integrating concepts from quantum computing with classical machine learning techniques, this research aims to develop a method that not only improves classification accuracy but also reduces computational requirements. The algorithm leverages singular value decomposition (SVD) and tensor networks to process and analyze astronomical images more effectively.

The primary objectives of this thesis are:

- To design and implement a quantum-inspired classification model.
- Evaluate the quantum-inspired classification model performance against traditional CNN-based methods.
- Demonstrate the quantum-inspired classification models potential in enhancing the efficiency of galaxy classification tasks.

The classification model, the compression algorithms and the code to calculate the used parameters in this project are released on the public GitHub corresponding to this project [3].

1.1 Thesis overview

Section 2 contains background information on subjects that are used in this project about quantum-inspired classification algorithms; Section 3 describes the methods used to research and define the problems in this thesis; Section 4 contains the results of the experiments carried out during the project; Section 5 contains the conclusion of the project and touches on further work that can be done on this subject.

2 Background information

2.1 Machine learning

Machine learning [4] is a form of Artificial Intelligence (AI) that focuses on building systems that can learn from the processed data or use data to perform better. This subject revolves around the development of algorithms and models that enable computers to recognize patterns, make predictions on these patterns and continuously refine their performance based on data inputs. Machine learning can also be applied to computer vision allowing for the classification of structures in images.

2.2 Computer vision

Computer vision [5] is an area of computer science that focuses on using computers to identify objects and people in images and videos. Like other types of AI, computer vision attempts to perform and automate human tasks. Computer vision focuses specifically on simulating human vision and the way people process what they see. It involves the development of algorithms and systems that empower computers to analyze and make decisions based on visual data, such as images or video.

2.3 Convolutional neural networks

Convolutional neural networks (CNNs) [6] are a subclass of machine learning and artificial neural networks which have the ability to determine particular features and patterns of a given input. Because of this, they are distinguished from other neural networks and are commonly used in image recognition. The capability of determining features is a result of the two types of layers used in a CNN, the convolutional layer and pooling layer. These layers are alternated to detect patterns and associate each pattern to a shape that is relevant to an image.

The convolutional layer makes use of a kernel, which determines features and patterns of a particular input [7]. It can associate these features with a given output in the training process, and uses this process to train the dataset. In contrast to that, a pooling layer reduces the dimensionality of the input data, reducing the number of parameters in the input and reducing the computational cost of the CNN [7].

2.4 Singular value decomposition

Singular value decomposition (SVD) is a matrix factorization technique used in linear algebra and data analysis [8]. It decomposes a $m \times n$ matrix into three simpler matrices, revealing the underlying structure and properties of the original matrix. Given a matrix A, singular value decomposition is represented as

$$A = U\Sigma V^T$$

Where:

- U is an $m \times m$ unitary matrix.
- Σ is an $m \times n$ diagonal matrix with non-negative real numbers on the diagonal, which are the singular values.
- V^T is the conjugate transpose of an $n \times n$ complex unitary matrix V.

These singular values are non-negative and describe the scaling factors applied to the rows and columns of the matrix A by the transformations represented by U and V. SVD has various applications, but most importantly for this project, image compression. Grayscaled images can easily be transformed into a matrix representation. The singular values in matrix Σ correspond to the square root of the eigenvalues obtained with matrix $A^T A$. From matrix Σ, information of the images can be resolved and used for image compression.

2.5 Tensor networks

Tensor networks are a mathematical framework used to represent and manipulate large multi-dimensional arrays and are used in quantum physics, machine learning, and various other fields to represent and manipulate high-dimensional data efficiently [9]. They involve tensors, which are multi-dimensional arrays of numbers, and the networks connecting these tensors according to specific rules. Tensor networks provide a structured way to approximate and analyze complex systems by decomposing them into simpler components.
In quantum physics, tensor networks are utilized to study entanglement and simulate quantum states. In machine learning, they offer a flexible framework for modeling and processing multi-dimensional data, such as images or sequences.

2.6 Related work

A notable contribution on the topic of computer vision using tensors would be the paper of Stoudenmire [10] on "Supervised Learning with Tensor Networks". This is a paper on tensor methods in machine learning. Here, the tensors are used to learn the MNIST dataset [11], which is a database of handwritten digits containing a training set of 60,000 examples, and a test set of 10,000 examples.

Furthermore, there is a paper that relates to the project description and data reduction of the Galaxy Zoo 2 (GZ2) database by Willet et al. [12]. This paper is on a citizen science project with more than 16 million morphological classifications of around 239000 galaxies drawn from the Sloan Digital Sky Survey. GZ2 uses classifications from volunteer citizen scientists to determine morphologies of galaxies. Stating that, while the original Galaxy Zoo 1 (GZ1) [13] project identified galaxies as early-types, late-types, or mergers, GZ2 defines finer morphological features. The full morphological classification is described in appendix A.

Lastly, there is the Medium publication by Thomas McRobie on "Applying a Deep Learning Approach to Galaxy Classification with Galaxy Zoo" [14]. This publication focuses on the difference in learning capability between the GZ1 database morphology and the GZ2 database morphology, concluding that the GZ2 morphological labels are much more useful than their GZ1 counterparts.

3 Method

3.1 Data analysis

The GZ2 dataset [15] briefly described in section 2.6 will be used for the experiments in this paper. As explained earlier, this is an improved version of the GZ1 database. When working on the classification algorithm, which would be needed for comparison to the eventual quantum-inspired algorithm, a problem occurred. Due to the fact that there were 818 classes, with some classes having very few items in them, was causing nearly no chance to correctly validate these classes. So after consideration it was decided to restrict it to a binary classifier of only spiralled and elliptical galaxies. The dataset contains 141.386 usable samples for spiralled classified galaxies and 97.643 usable samples for elliptical classified galaxies.

To reduce runtime of the classification algorithm it was useful to experiment with grayscaling the image as, presumably, the intensity and spiraled structure of a galaxy should not rely on the coloration of the image in the visible spectrum. The use of grayscaling is shown in figure 1. The main advantage of having the images in black and white is that for every image only one third of the information of the original image is kept, resulting in reduced runtime.

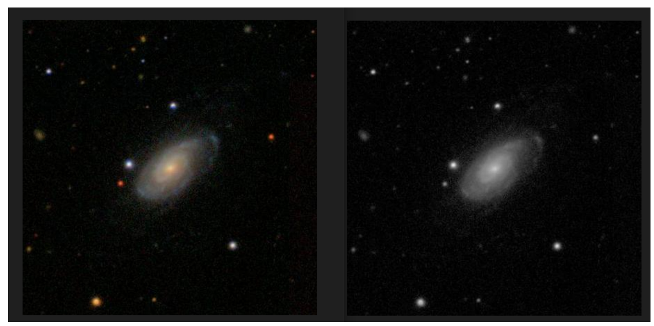

Figure 1: *Example of grayscaled image (right) of a spiraled galaxy (left).*

To justify using a complex classification algorithm it should not be clear from looking at geomorphological features of the image to predict which class is looked at. To this end the mean and variance of the intensity of each image was calculated. For both the mean and the variance the classes were compared to each other. If these parameters contain a lot of similar values it is not clear, when looking at these values alone, which class the specific image is in.

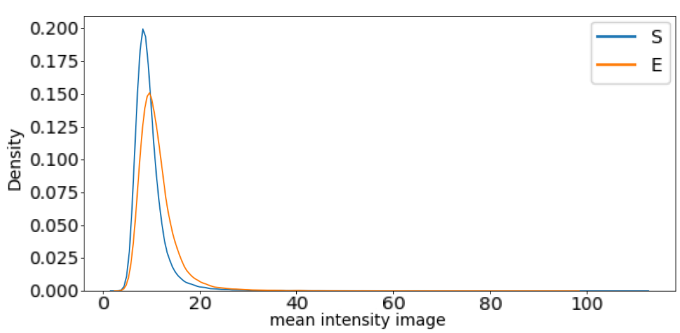

Figure 2: *Kernel density estimate plot of the mean intensity of typical galaxies.*

The mean was calculated by taking the sum of intensity per pixel divided by the total number of pixels in the image [16]. To get a visual representation of the density per intensity of typical galaxies, a kernel density estimate plot was made. This is the application of kernel smoothing for probability density estimation, so a non-parametric method to estimate the probability density function of a random variable based on kernels as weights [17]. The result can be seen in figure 2.

In this figure the E stands for the elliptical galaxies and the S for spiralled galaxies. From figure 2 can be seen that the spiralled galaxies have more chance to have a lower overall intensity, but that the distributions generally overlap so that calculating the mean of an image does not provide clear information about the class.

The variance was calculated by measuring how far each intensity is from the mean for every pixel in the image [16]. This is again plotted in a kernel density estimate plot to help visualize this metric and can be seen in figure 3.

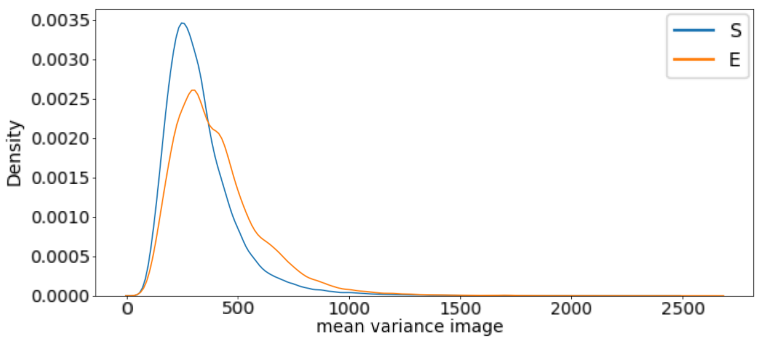

Figure 3: *Kernel density estimate plot of the intensity variance of typical galaxies.*

From figure 3 it can be seen that elliptical galaxies have a larger variance, which can be explained after looking at, for example, figure 4, which shows an elliptical galaxy surrounded by a haze of light. This most likely causes these higher variances in the image, but again that the distributions generally overlap so that calculating the variance of an image does not provide clear information about the class. So it would not really be possible to distinguish galaxies solely on these factors, hereby justifying the need for a classification algorithm.

3.2 Classification accuracy baseline

The next step consists of setting up a baseline accuracy to compare with the results of the classification algorithm. In the GZ2 database, the authors have added vote fractions [15] to each entry containing information on the fraction of people that voted if a specific image was featureless, thus an elliptical classified galaxy, or if a galaxy contained a feature/disk, thus being classified as a spiraled galaxy.

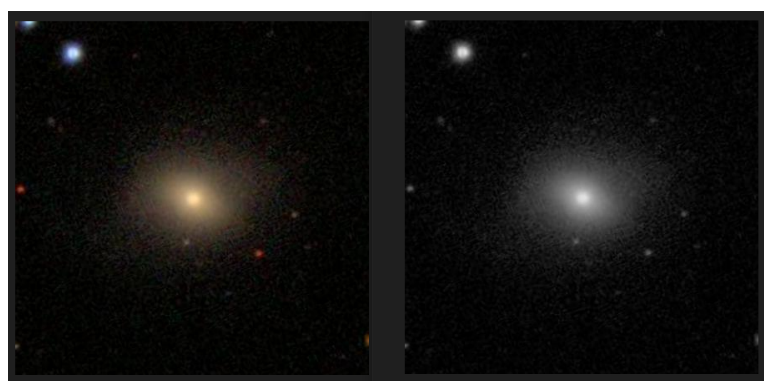

Figure 4: *Example of grayscaled image (right) of an elliptical galaxy (left).*

By averaging the vote fraction per class on entries that were actually labeled that class, you would get the following percentages: the average percentage of people that voted that the images they saw were elliptical galaxies that were actually classified as elliptical galaxies was 85,3%; the average percentage of people that voted that the images they saw were spiraled galaxies that were actually classified as spiraled galaxies was only 47,1%.

From this can be seen that on average only around 47% of the people who voted on a spiraled galaxy were correct in their judgement on the galaxy being classified as spiraled. The authors of the study assume that the reason for this is that a bias is present on the voting system for the galaxies [18]. This is because a large amount of the galaxies in the dataset are more distant galaxies, which are on average, both smaller and dimmer in the cutout images. This leads to a result where finer morphological features are more difficult to identify. When looking through the images it becomes more apparent what this means, as can be seen in figure 5. These galaxies are classified as spiraled galaxies but show no obvious signs of these features.

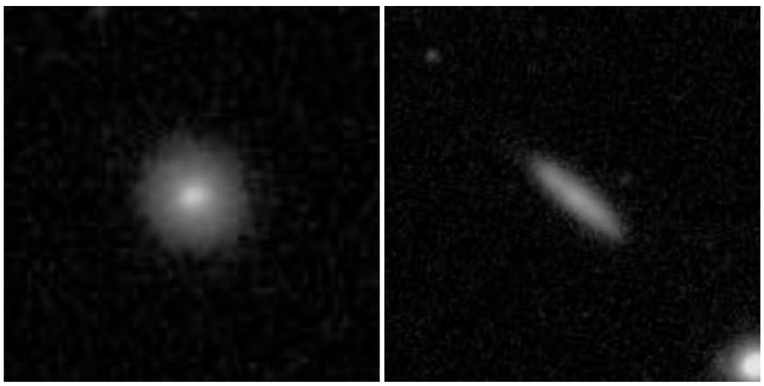

Figure 5: *Examples of spiraled classified galaxies without clear signs of features.*

They compensate this bias by using the spectroscopic redshifts to determine a better debiased fraction based on "the assumption that for a galaxy of a given physical brightness and size, a sample of other galaxies with similar brightnesses and sizes will share the same average mix of morphologies." [12].

The average debiased percentage of people that voted on elliptical galaxies that were actually classified as elliptical galaxies was only 76,1%, the average debiased percentage of people that voted on spiraled galaxies that were actually classified as spiraled galaxies was 86,2%.

So on average, the biased percentage of votes that was correct was 66,2%, and the average debiased percentage of votes was 81,2%. Since the debiasing method of the GZ2 paper is based on an assumption, which perhaps holds no real precedent, the classification accuracy baseline for this project will be 66,2%.

3.3 Classification model

The classification model used for this thesis was made with the Keras [19] open source library in Python [20]. It features a convolutional neural network using 2 convolutional layers which are both followed by pooling layers.

The convolutional layers are used to extract features from the input image and the pooling layers are used to reduce the spatial dimensions of the feature maps while retaining the most important information. The flatten layer flattens the output of the convolutional layer to a single array followed by one dense layer which adds extra outputs. Finally, an L2 regulizer is used to prevent the model achieving overfitting on the training dataset. A summary of the model used for the experiments in this thesis is found in figure 6.

4 Results

In this section the results of the experiments performed in this thesis will be discussed. In total, 4 experiments were conducted. The first was to look if the dataset containing only the grayscaled images could be learned correctly by the CNN model and then comparing it to the baseline, which is described in section 4.1. The second experiment was to compress the dataset using singular value decomposition and comparing that to both the uncompressed dataset and the baseline, The results of the second experiment are described in section 4.2. The third experiment compares the learnability of one of the SVD compressed datasets to that of the original dataset. These results are shown in section 4.3. And finally, in the last experiment the dataset was compressed again, but this time the images were first loaded into tensors before being compressed using SVD. These results are then compared to the previous experiments and can be found in section 4.4.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
module_wrapper (ModuleWrappe (None, 424, 424, 3)       0
_________________________________________________________________
conv2d (Conv2D)              (None, 415, 415, 32)      9632
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 207, 207, 32)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 188, 188, 16)      204816
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 9, 9, 16)          0
_________________________________________________________________
flatten (Flatten)            (None, 1296)              0
_________________________________________________________________
dense (Dense)                (None, 2)                 2594
=================================================================
Total params: 217,042
Trainable params: 217,042
Non-trainable params: 0
_________________________________________________________________
```

Figure 6: *Convolutional neural network model summary.*

4.1 Classic convolutional neural network

First, the model was trained on images of the GZ2 dataset. These are images of 424 x 424 pixels and were grayscaled to improve time complexity. The model was allowed to be trained over 8 epochs and resulted into a validation accuracy of 79,4%. Comparing to the baseline set in the previous paragraph shows that our model outperformes the baseline of 66,2% by a substantial amount. It can also be seen that it almost reaches the debiased average percentage by still using the original images, and not looking at spectroscopic redshifts attached to these images.

4.2 Singular value decomposition data

For comparison with later experiments using compression of the dataset, it was useful to test the model using SVD compressed images. The variance of the top 20 singular values of each decomposed image is an important benchmark of information that is conveyed in these images. By reducing the amount of eigenvectors used for information, the image can be compressed into a perhaps less detailed image but smaller file size. The first 20 singular eigenvectors corresponding to the first 20 eigenvalues of the images averaged over the dataset can be seen in figure 7.

To test the learning capability of the model we had to compress the SVD compressed data at certain value decomposed points. This would then be compared to the original model as a benchmark. The images were compressed by using only the first eigenvector and the first 5, 10 15 and 20 eigenvectors, thus using only these first few eigenvalues. Also, an image was 'compressed', but instead using all 424 eigenvectors corresponding to the height and width of the original image. This image would of course look exactly the same as the original image. An example of these images can be seen in figure 8.

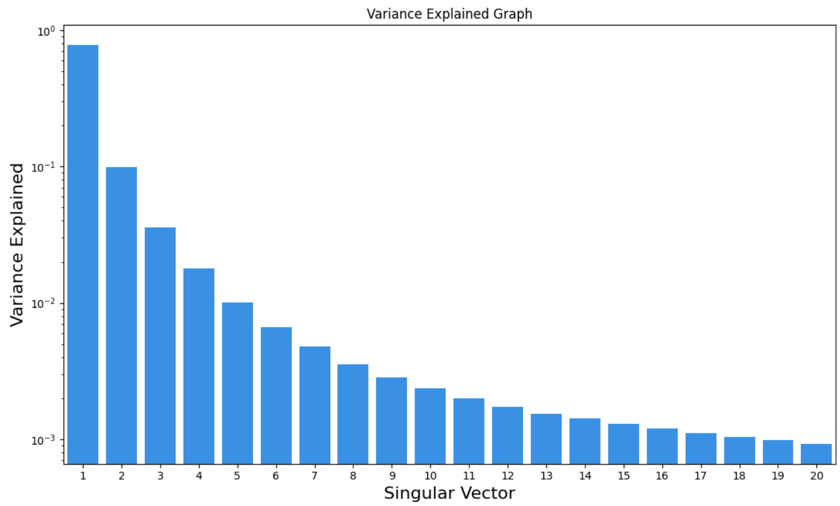

Figure 7: *Logarithmic plot of first 20 singular eigenvectors corresponding to the first 20 eigenvalues averaged over the dataset.*

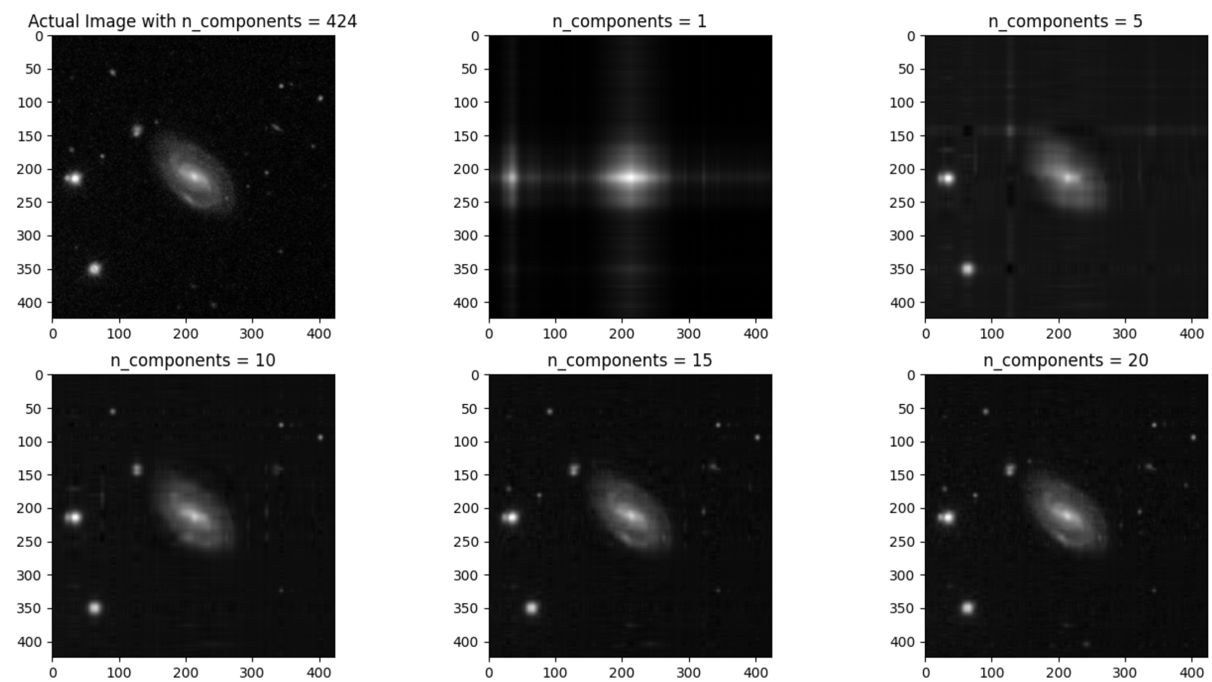

Figure 8: *Galaxy images SVD compressed with all(top left), first(top middle), and the first 5(top right), 10(bottom left), 15(bottom middle) and 20(bottom right) eigenvectors.*

As can be seen from figure 8, the compressed image containing the information of only the first eigenvector produces only a bright spot where the galaxy should have been displayed in the original image. The first 5 eigenvectors shows an image that somewhat resembles a galaxy. The detail start to show up when using the first 10 eigenvectors and it would be possible to conclude that this image contains a spiraled galaxy. The details keep getting better when increasing the amount of eigenvector information to the image, with the 20 first eigenvector image almost looking exactly the same as the original image.

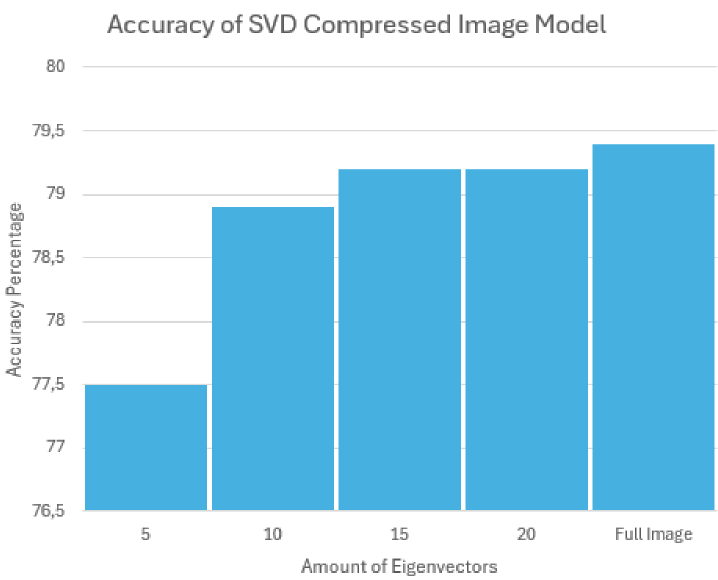

Figure 9: *Average model accuracy on the SVD compressed image datasets and the full image dataset.*

Since the data with only the first eigenvector will not produce any insightful images it will be left out of the classification model, as most likely the accuracy will be nowhere near that of the other compressed datasets.

With the same model, each dataset was then learned over 8 epochs resulting in the graph in figure 9. As can been seen, the accuracy of the model increases as the details of the compressed images increases, with finally the first 15 and 20 eigenvectors being only 0,2% lower than the accuracy of the model on the full images.

4.3 Comparison model learnability 15 eigenvectors and full image

Another experiment worth doing was to look if a downsized version of the classification model could learn the dataset that uses only the first 15 eigenvectors, but would not be able to learn properly on the dataset containing the original gray scaled images. A model was made that uses half of the convolutional and pooling layers that the original model had. A summary of this model can be seen in figure 10.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)           (None, 415, 415, 32)      9632
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 20, 20, 32)       0
_________________________________________________________________
flatten_1 (Flatten)         (None, 12800)             0
_________________________________________________________________
dense_1 (Dense)             (None, 2)                 25602
=================================================================
Total params: 35,234
Trainable params: 35,234
Non-trainable params: 0
_________________________________________________________________
```

Figure 10: *Downscaled model summary.*

The two datasets were then learned on the model and gave some interesting results. The dataset that was compressed using only the first 15 eigenvectors could be learned consistently at around baseline classification accuracy with a percentage of 69,3%. The dataset containing the uncompressed images however would half of the time get stuck in a suboptimal minimum, resulting in accuracies below 60%, when it did not get stuck in a suboptimal minimum, the model would achieve a performance of around 71%. This resulted in an average accuracy just above baseline prediction of 66,2%.

4.4 Tensor SVD compressed data

A similar method of performance measuring as for singular value decomposition was used on the dataset that was put into tensors and then truncated with the SVD method. Again, the images were compressed by using only the first eigenvector and the first 5, 10, 15 and 20 eigenvectors. Again an image was 'compressed' also using all 424 eigenvectors corresponding to the height and width of the original image, resulting in an image looking exactly the same as the original image. An example of these images can be seen in figure 11.

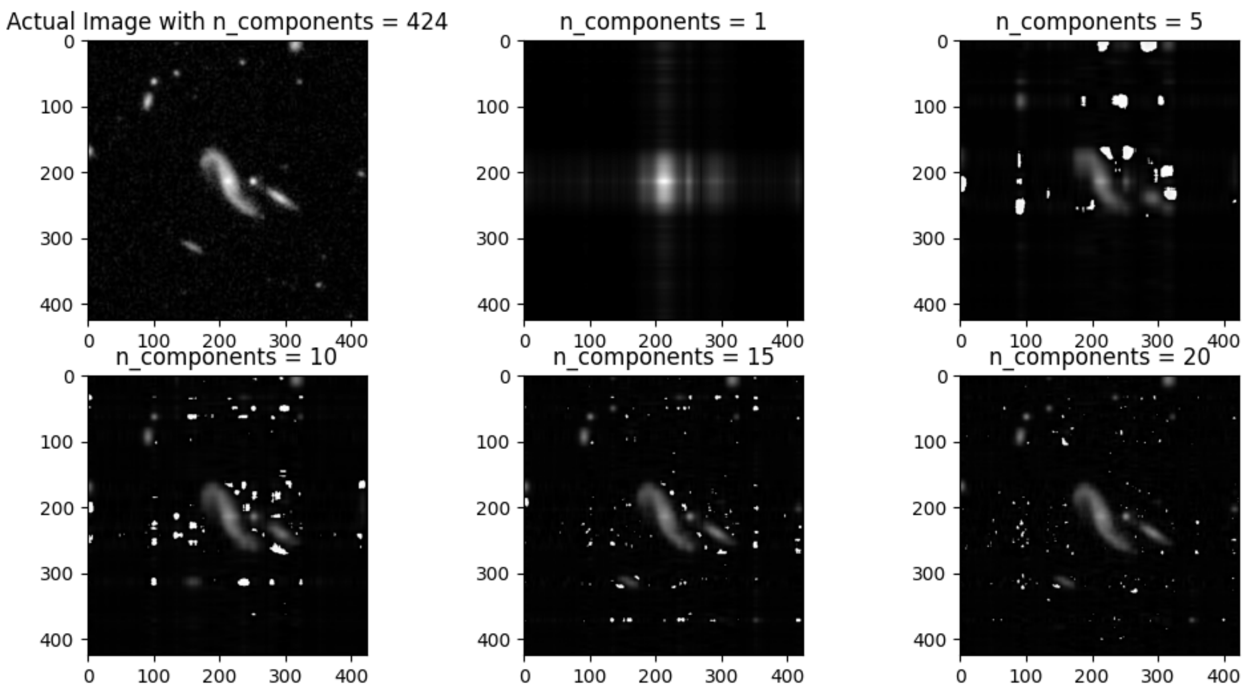

Figure 11: *Galaxy images tensor SVD compressed with all(top left), first(top middle), and the first 5(top right), 10(bottom left), 15(bottom middle) and 20(bottom right) eigenvectors.*

An obvious difference between the ordinary singular value decomposition and the decomposition on the tensor is that the tensor SVD has a lot of issues with noise from the original image. In images compressed with only a few eigenvectors these specks are large but only few. When increasing the eigenvectors, the noise multiplies and becomes smaller.

With the same model, each dataset was then learned over 8 epochs resulting in the graph in figure 12. As can been seen, the accuracy of the model increases as the detail of the compressed images increases. But compared to the classically singular value decomposed images, the classification accuracy becomes worse. With the most detailed test case, the tensor compressed images with the first 20 eigenvectors, a classification accuracy of only 78,0% was reached

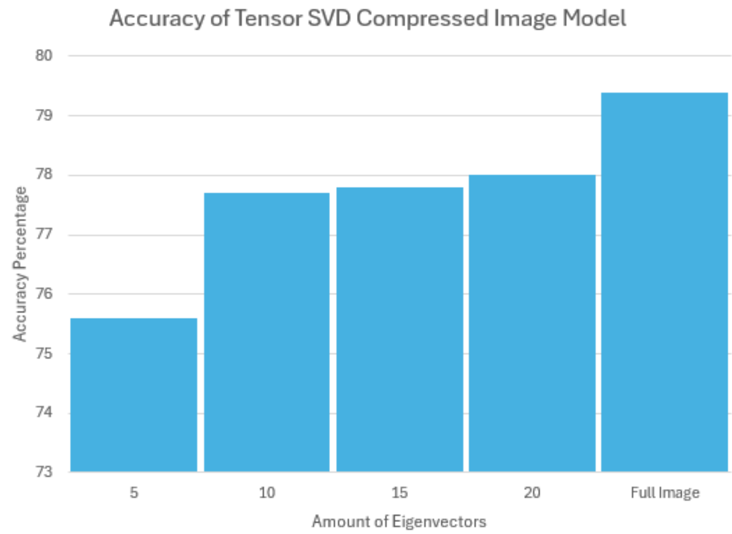

Figure 12: *Average model accuracy on the tensor SVD compressed image datasets and the full image dataset.*

4.5 Highest accuracy per dataset

As shown in previous sections, the model could sometimes converge in a suboptimal maximum or become overfitted, bringing the average accuracy down in the previous experiments. So it would also be interesting to look at the best accuracy that the model achieved while validating each dataset. On the original grayscaled image dataset the best accuracy was an accuracy percentage of 80,1% achieved on the 5th epoch.

For singular value decomposition the model achieved higher accuracy when trained on the dataset that was truncated with only 15 and 20 eigenvectors in both cases, but only managed to get there on average on epoch 6, with respectively achieving 80,3% and 80,2%. The datasets that were compressed with less eigenvectors decreased in accuracy when compared to those with a larger amount of eigenvectors, but both reached these accuracies at earlier epochs during training the model. The dataset truncated with 5 eigenvectors performed with an accuracy of 78,4% at epoch 5. The dataset truncated with 10 eigenvectors performed with an accuracy of 79,7% at epoch 3.

Overall, the tensor SVD compressed dataset has a lower maximum accuracy than that of the SVD method. Another thing to mention is that all tensor SVD compressed datasets performed at their best at a later stage of training compared to the experiments with the original dataset and the regular SVD compressed dataset. The dataset with 5 eigenvectors performed with an accuracy of 77,4% at epoch 7. The truncated dataset with 10 eigenvectors performed with an accuracy of 79,3% at epoch 8. The dataset with 15 eigenvectors performed with an accuracy of 79,1% at epoch 6. The dataset with 20 eigenvectors performed with an accuracy of 79,3% at epoch 6.

5 Conclusions and further research

The main goal of the thesis was to find out if it was possible to make a quantum-inspired algorithm for galaxy classification.

A benchmark was proposed in section 3.2 to check the effectiveness of the model, since the dataset did not originally come with one and it could also be used in further research by allowing other compression models to be compared to this benchmark.

From the results it can be seen that, in comparison to this classification accuracy baseline, the convolutional neural network defined in section 3.3, could get to a much better classification accuracy when trained on the dataset containing the full sized imagery. Averaged over the full training duration the classification model achieved a 13,2% better classification accuracy than the baseline that was set in section 3.2.

From the experiment on singular value decomposition from section 4.2 it can be concluded that the SVD truncated images, averaged over the full training duration, could almost get to the same level of classification accuracy as the dataset containing the full sized imagery.

In contrast, the compressed data that was truncated using only the first 15 and 20 eigenvectors only reached a 0,2% lower accuracy than the of the model on the full image. This means that, averaged over the full training duration, the classification model learning on the 15 and 20 eigenvectors truncated dataset achieved a 13% better classification accuracy than the baseline set in section 3.2. The 5 and the 10 eigenvector truncated dataset achieved respectively a 11,3% and 12,7% better classification accuracy than the baseline.

However this compression algorithm shows that even with only a fraction of the information of the original image, the classification algorithm could still achieve a higher classification accuracy than the baseline set in section 3.2. This means that a significant data reduction can be achieved on the original dataset while still being more accurate than the classification accuracy baseline.

The main problem with the tensor SVD compressed data can be seen in the results from the experiment in section 4.4, as the compression method used causes noise to appear in the images. This could also be the cause of the neural network not correctly learning in the same capacity, as the model does improve on the original and the SVD compressed dataset.

Overall the tensor truncated datasets performed worse than the full sized imagery dataset when learned by the classification model, but still achieved better classification accuracy than the baseline set in section 3.2. The datasets truncated with 10, 15 and 20 eigenvectors achieved around the same classification accuracy and performed 11,5%, 11,6% and 11,8%, respectively, better than the classification accuracy baseline. The worst performing dataset was the tensor truncated dataset using only 5 eigenvectors with still a 9,4% better classification accuracy than the baseline.

Again this compression algorithm achieves a significant data reduction compared to the original dataset while still being more accurate than the classification accuracy baseline. The exceptions to this conclusion are the 15 and 20 eigenvectors tensor SVD truncated dataset. These datasets have a higher data size compared to the original dataset due to the increase in randomly located noise in the images.

In conclusion, it can be seen that the dataset that was compressed by quantum-inspired algorithms is harder to learn than the original dataset for this model. It also performs a little bit worse compared to the dataset compressed using traditional singular value decomposition. All of these methods do, however, perform better that the baseline, which means that it can identify galaxy classes better than a normal person could.

Based on the results presented in this paper, it can be concluded that it is possible to create a classification model using a quantum-inspired algorithm. However it does not perform as good as singular value decomposition for usage as a compression algorithm.

5.1 Further Research

For further research it would be interesting to investigate the performance of the model on datasets that are compressed with different algorithms such as JPEG (using dicrete cosine transform) or PNG (using color quantization), and use the classification benchmark proposed in this thesis to check their classification accuracy.

Furthermore, it could be interesting to check the model, and the compression algorithms, used in this thesis to classify other types of astronomical objects, such as stars, quasars, and nebulae. This would test the versatility and robustness of the algorithm across different datasets and classification tasks.

Finally, it could be interesting to compare the quantum-inspired algorithm in this thesis with other advanced machine learning models, to benchmark its performance and identify areas for improvement.

References

[1] L. Chao, Z. Wen-Hui, L. Ran, W. Jun-Yi, and L. Ji-Ming, "Research on Star/Galaxy classification based on stacking ensemble learning," Chinese Astronomy and Astrophysics, vol. 44, no. 3, pp. 345–355, Jul. 2020, doi: 10.1016/j.chinastron.2020.08.005.

[2] "Webb Image Release- Webb Space Telescope GSFC/NASA." (Accessed: 03-06-2024) URL: https://webb.nasa.gov/

[3] Max Veraar. A quantum-inspired algorithm for galaxy classification. (Accessed: 25-06-2024) URL: https://github.com/MaxVeraar/A-quantum-inspired-algorithm-for-galaxy-classification

[4] IBM. What is machine learning (ML)? (Accessed: 17-04-2024). URL: https://www.ibm.com/topics/machine-learning

[5] IBM. What is computer vision? (Accessed: 18-04-2024). URL: https://www.ibm.com/topics/computer-vision

[6] IBM. What are convolutional neural networks? (Accessed: 18-04-2024). URL: https://www.ibm.com/topics/convolutional-neural-networks

[7] Q. Ke, J. Liu, M. Bennamoun, S. An, F. Sohel, and F. Boussaid, "Computer Vision for Human–Machine interaction," in Elsevier eBooks, 2018, pp. 127–145. doi: 10.1016/b978-0-12-813445-0.00005-8.

[8] D. C. Lay, S. R. Lay, and J. McDonald, Linear Algebra and its Applications (Fifth Edition). pp. 432-448, 2020.

[9] J. Biamonte and V. Bergholm, "Tensor networks in a nutshell," arXiv (Cornell University), Jan. 2017, doi: 10.48550/arxiv.1708.00006.

[10] E. M. Stoudenmire and D. J. Schwab, "Supervised Learning with Tensor Networks," Neural Information Processing Systems, vol. 29, pp. 4799–4807, Jan. 2016, [Online]. Available: https://papers.nips.cc/paper/6211-supervised-learning-with-tensor-networks.pdf

[11] "THE MNIST DATABASE of handwritten digits". Yann LeCun et al. (Accessed: 16-04-2024) URL: http://yann.lecun.com/exdb/mnist/

[12] K. W. Willett et al., "Galaxy Zoo 2: detailed morphological classifications for 304 122 galaxies from the Sloan Digital Sky Survey," Monthly Notices of the Royal Astronomical Society, vol. 435, no. 4, pp. 2835–2860, Sep. 2013, doi: 10.1093/mnras/stt1458.

[13] C. J. Lintott et al., "Galaxy Zoo: morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey," Monthly Notices of the Royal Astronomical Society, vol. 389, no. 3, pp. 1179–1189, Sep. 2008, doi: 10.1111/j.1365-2966.2008.13689.x.

[14] T. McRobie, "Applying a Deep Learning Approach to Galaxy Classification with Galaxy Zoo," Medium, Jul. 29, 2023. [Online]. (Accessed: 13-03-2024) URL: https://medium.com/@thomas.mcrobie999/applying-a-deep-learning-approach-to-galaxy-classification-with-galaxy-zoo-2-afb51c81541f

[15] K. W. Willett et al., "Galaxy Zoo 2: Images from Original Sample," Zenodo (CERN European Organization for Nuclear Research). Nov. 01, 2013. doi: 10.5281/zenodo.3565489.

[16] A. Agresti, Statistical Methods for the Social Sciences, Global Edition. Pearson Higher Ed, 2018.

[17] S. Weglarczyk, "Kernel density estimation and its application," ITM Web of Conferences, vol. 23, p. 00037, Jan. 2018, doi: 10.1051/itmconf/20182300037.

[18] R. E. Hart et al., "Galaxy Zoo: comparing the demographics of spiral arm number and a new method for correcting redshift bias," Monthly Notices of the Royal Astronomical Society, vol. 461, no. 4, pp. 3663–3682, Jul. 2016, doi: 10.1093/mnras/stw1588.

[19] Keras-team. Keras - open-source Python library. (Accessed: 22-05-2024). URL: https://keras.io/

[20] Python Software Foundation. Python - programming language. (Accessed: 22-05-2024). URL: https://www.python.org/

A Generating the abbreviation for a GZ2 morphological classification

As part of the GZ2 data release [12], a short abbreviation (gz2 class) is provided that indicates the most common consensus classification for the galaxy. It is emphasised by the authors of the GZ2 paper that the intent is not to create a new classification system; rather, this is only a convenient shorthand for interpreting portions of the GZ2 results. The gz2 class string is generated for each galaxy by taking the largest debiased vote fraction and selecting the most common response for each subsequent task in the decision tree. Galaxies that are smooth have gz2 class strings beginning with 'E'. Their degree of roundness (completely round, in-between, and cigar-shaped) is represented by 'r','i', and 'c', respectively. Galaxies with features/disks have gz2 class strings beginning with 'S'. Edge-on disks follow this with 'er', 'eb', or 'en' (with the second letter classifying the bulge shape as round, boxy, or none). For oblique disks, the letter following 'S' is an upper-case 'B' if the galaxies have a bar. The bulge prominence ('d' = none, 'c' = just noticeable, 'b' = obvious, 'a' = dominant). Both bars and bulges follow the same general trends as the Hubble sequence, although the correspondence is not exact. If spiral structure was identified, then the string includes two characters indicating the number (1, 2, 3, 4, +, ?) and relative winding ('t'=tight, 'm'=medium, 'l'=loose) of the spiral arms. Finally, any feature in the galaxy the users identified as "odd" appears at the end of the string in parentheses: '(r)'=ring, '(l)'=lens/arc, '(d)'=disturbed, '(i)'=irregular, '(o)'=other, '(m)'=merger, '(u)'=dust lane. Objects that are stars or artifacts have the gz2 class string 'A'. A few examples of gz2 class strings would be:

- Er = smooth galaxy, completely round
- SBc2m = barred disk galaxy with a just noticeable bulge and two medium-wound spiral arms
- Seb = edge-on disk galaxy with a boxy bulge
- Sc(I) = disk galaxy with a just noticeable bulge, no spiral structure, and irregular morphology
- A = star

Sample images of the twelve most common GZ2 class labels are shown in Figure A.1.

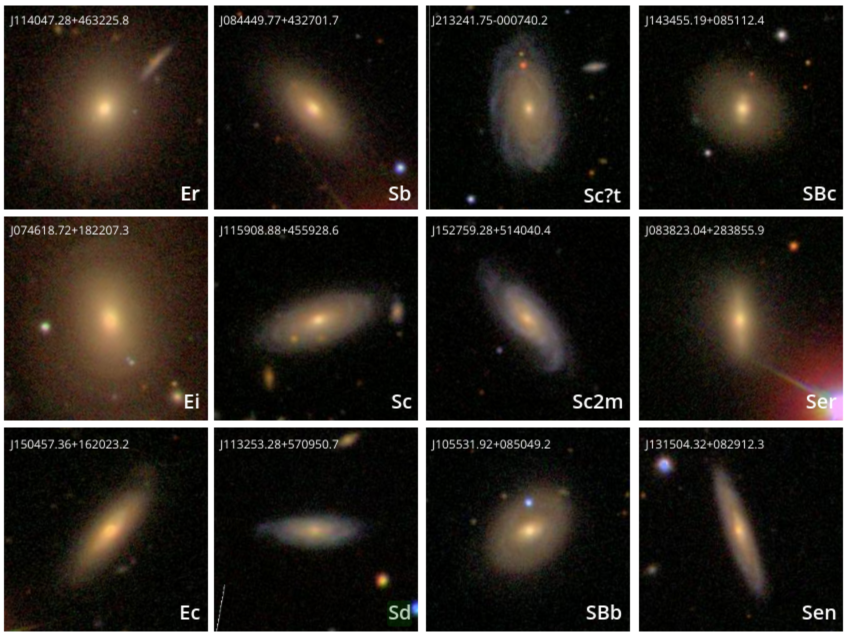

Figure A.1: Example images with their GZ2 classifications.