# Convolutional Autoencoder for detecting outlier in the Sloane Digital Sky Survey

Ross Erskine (ppxre1)[1, *]
[1]*School of Physics and Astronomy, University of Nottingham, Nottingham, NG7 2RD, UK*
(Dated: December 14, 2022)

The Sloan Digital Sky Survey (SDSS) is a international collaboration that has mappped up-to one-third of the night sky with over three million observations of Galaxies, Stars, and Quasars with multi-colour imagery, over the last 20 years. The discovery of outliers proposes new opportunities in exploring unknown phenomena. In this paper we propose a Convolutional autoencoder (CAE); by just using the encoder part of the encoder and using kernel density estimation (KDE) to find outliers in the SDSS dataset. The CAE model managed to find four outliers in a batch of 64, however more rigorous testing is needed to understand the model further.

## I. INTRODUCTION

The Sloan Digital Sky Survey (SDSS) is a international collaboration that has mappped up-to one-third of the night sky with over three million observations [1] of Galaxies, Stars,and Quasars with multi-colour imagary, over the last 20 years. With the next phase SDSS-V on the horizon with the next data release in December 2022, researchers will want to exlore more and more data. The discovery of outliers proposes new opportunities in exploring unknown phenomena and can produce some of the weirdest Galaxies in the universe [2], helping us understand more and more about what can possibly be out there. Convolution autoencoder (CAE) architecture is generally used as a dimensionality reduction technique or feature learning. Although can also be used as a unsupervised method that can be trained to encode input features into smaller dimensional space, then reconstruct the features in to a reduced dimensional space, allowing us to measure discrepancies from the original and identify outliers.

Outlier's are objects that do not fit the model and using unsupervised learning methods to find them is more proactive than supervised methods [3]. The SDSS objects are classified based on the difference of cmodel and PSF magnitude [4]. It is natural at this point to analyse clusters for outlier detection, searching for objects in the tail of well characterised distributions, which could also offer an opportunity to unveil new phenomena or the "unknown unknowns" [2]. Although there is not a great deal written on outliers using CAE, there is plenty of research written on outliers especially on SDSS data set such as: [3] compared different types of unsupervised algorithms for example *local outlier factor*, *isoloation forest*, *k-means clustering* and CAE; finding that most algorithms found bright objects to be outliers with 80% being stars and 20% being galaxies. [5] used a CNN autoencoder (CNNAE) on wavelengths of *spectra*, [6] and [7] used Local outlier factor algorithm distributed on large number of spectra. While[8] used *object cluster analysis* (OCA) and *self organising maps* (SOM) to analyse outliers, [9] created a *deep generative network* (GAN) and simulated their own galaxy data to help find outliers. [10] used *K-nearest neighbour* algorithm; training on *Kepler* data, while also testing algorithm using Boyajian's star that is a notorious anomaly. Finally [2] used unsupervised *Random forest* algorithm that was tested on more two million galaxy spectra and discussing in depth at the possibilities of their findings. One issue with finding outliers in large data sets is the run time of having to check every image and see if it counts as an outlier compared to the rest. One solution to this problem is *kernel density estimation* (KDE) by exploiting the reduced dimensional space to build a target *probability density function* [11]. We take a threshold from the tail end of the distribution allowing us to find outliers without going through the entire data set.

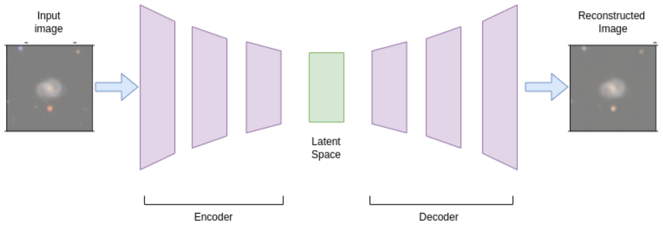

FIG. 1. A Convolutional Autoencoder composed of two parts an *encoder* with convolutional layers that extracts predominant features, which shrinks in size depending on the size of the *kernel*, creating a decompressed latent space representation, and a *decoder* with convolutional layers decompresses the latent space representation back into the original image extracting irrelevant features.

## II. METHODOLOGY

Convolutions are based on the *LeNet-5 network* [12], which a *kernel* or *filter* is applied to each colour channel extracting the most predominant features to create a new *feature map*. This naturally shrinks the size of the input, although *padding* can be added if the size of the input is to be attained [13]. Convolutional Autoencoders (CAE) is an unsupervised learning algorithm, which is

[*] ppxre1@nottingham.ac.uk

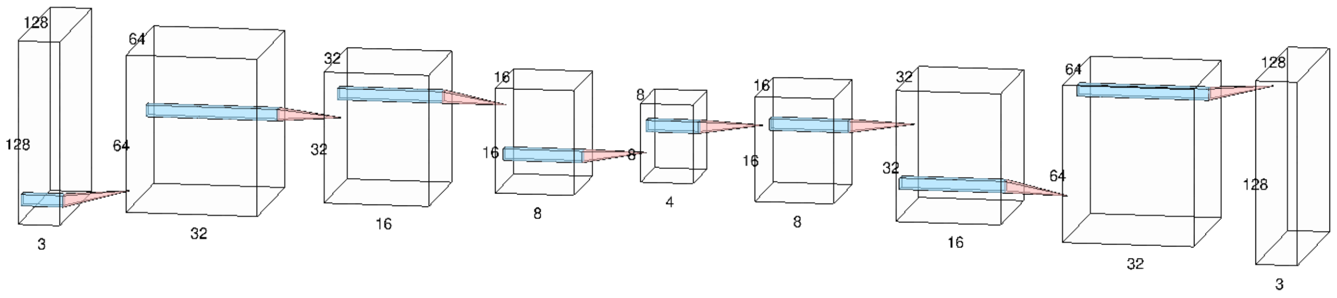

FIG. 2. The AE is composed of two parts A: *encoder* that starts with dimensions $128x128x3$, with each *kernel* being dimensions $3x3$ with a stride of 2. the first layer on the *encoder* the convolutions output is $64x64x32$, layer 2 output is $32x32x16$, layer 3 output is $16x16x8$ and finally the fully compressed image resides at $8x8x4$ giving a reduced total dimensionality with 256 total parameters. B: *decoder* is a mirror of the *encoder* that starts with input of $8x8x4$ onto the second layer with output of $16x16x8$ and so on until we get to our original dimensions of $128x128x3$.

composed of two parts an *encoder* and a *decoder*. The *encoder* $h = f(x)$ compresses the input into a smaller dimensional latent space representation or hidden layer $h$ using convolutions, which extracts the most relevant features. The *decoder* $g(f(x)) = x'$ decompresses that latent space representation back up using convolutions into the original image [14] Figure 1. The CAE algorithm learns by maximising the information and minimising the *reconstruction error* or *loss* function $L(x, g((f(x)))$ which in this case is the mean-squared error Equation1.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{1}$$

The architecture is based on work carried out by [15]that created a CAE model on the MNIST data set where the CAE was a $28x28$ diameter images, although the images used from [16] could have been up to $450x450$ diameter we opted for an input size of $128x128x3$ due to memory restrictions. We also decided to split our CAE into two seperate classes as we would like to use the *encoder* part alone to create our KDE later on; the two version where based on work from [17]. Each *kernel/filter* used is $3x3$ with a stride of 2, the first layer on the *encoder* using convolutions outputs a dimension $64x64x32$, layer 2 output is $32x32x16$, layer 3 output is $16x16x8$ and finally the fully compressed image resides at $8x8x4$ giving a reduced total dimensionality with 256 total parameters. The *decoder* is a mirror of the *encoder* that starts with input of $8x8x4$ onto the second layer with output of $16x16x8$ and so on until we get to our original dimensions of $128x128x3$ Figure 1.

The original galaxy zoo images from [16] has up to 250 thousand images we are only going to use 20 thousand for this experiment, training will be done in batches of 64 and 1000 epochs, with a validation set of batches of 32 to see whether the generalisation gap is minimal. This will ensure that the reduced dimensionality will have the best chance to represent our model.

After our training on the CAE has finished we will then attempt to create our PDF from a single batch from the training data set. Figure 3 shows the pipeline where we use just the encoder part of our CAE by passing each image from a single batch through the encoder we achieve our reduced latent space representation which was dimension $8x8x4$, then we flatten to a single vector of 256 length and using a KDE Equation 2; where $\alpha$ is constant normalisation factor and $h$ is the kernel bandwidth [11]. this produces or smoothed over PDF, which we can then take our thresholds from tail end of the distribution. This allows us to then test our model with either a single image or a batch of images, to classify if our model believes that the image is an outlier or not.

The model will be tested on the validation set that was used to minimise the generalisation gap. It will be interesting to see what images the test set will provide. [3] discusses finding images with lines on that are likely to be asteroids or satellites, however there research generally found that most outliers of stars and galaxies to be brighter than normal objects.

$$\overline{f}_X(x) = \frac{1}{\alpha}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \tag{2}$$

## III. RESULTS

The training loss see appendix: Figure 8 seem to suggest that the model converged after about 500 epochs meaning we probably did not need as many as a 1000 epochs to train, nothing about the training suggests that we would need more than 20000 images to train, however more images would take longer to converge perhaps giving better results. After testing our data on a sample batch from the validation set of data not seen during training. The results from the batch of an average density

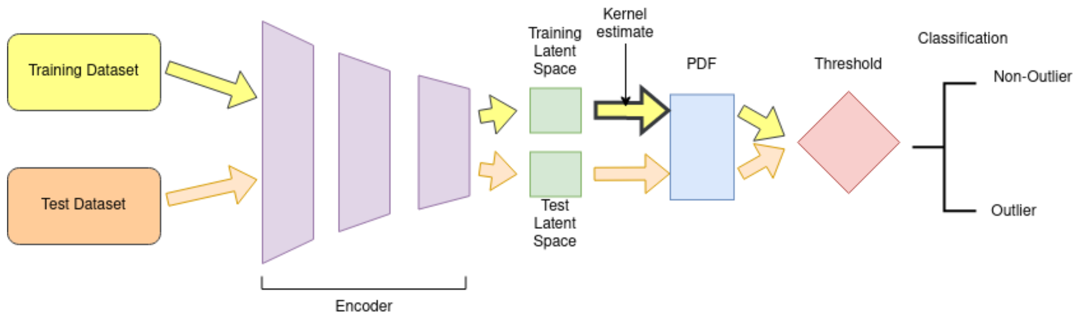

FIG. 3. The pipeline for creating a PDF using just the encoder or our CAE and KDE, which will give us our threshold for classifying test images as outliers or not.

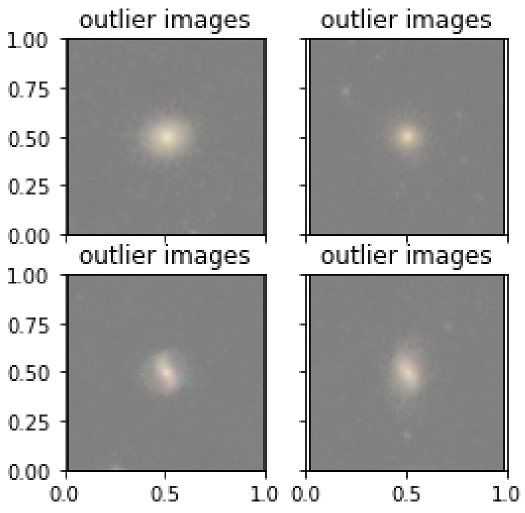

FIG. 4. Shows the four images that CAE model from a sample of the validation set suggests as outliers.

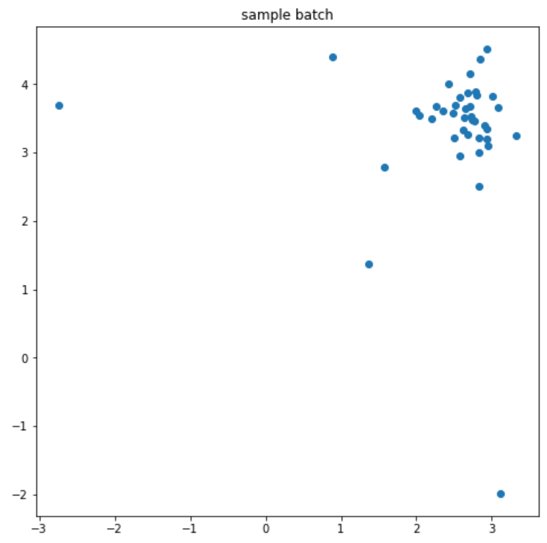

FIG. 5. Shows the encoded sample space proved by the encoder.

of -9727, standard deviation density of 11755, an average reconstruction error of 0.0007 and a standard deviation reconstruction error of 0.0003. Our model seem to suggest out of the sample batch 32 images that 4 of them where outliers see Figure 4. However nothing visually stands out such as lines from asteroids or satellite's or very bright objects [3] or merging galaxies [9], Although according to the encoded data from the validation batch sample see Figure 5 that you could believe that 4 objects stand out as outliers.

Looking at further work: time restraints meant the model was trained on only on 20000 images this could be increased give a better training and better representation of encoding, also more layers could be added such as dense layers, linear layers [17], or max-pooling layers [18]. However, outlier detection does seem to have a improved results when using spectra rather than actual image [5], [6], [8]. Also increased tests perhaps with known outliers such as *Boyajian's* star [10] to help understand the CAE model more, and more analysis between different densities obtained from the model.

## IV. CONCLUSIONS

We proposed a method of using *Convolutional autoencoder*CAE to find outliers in the SDSS galaxyZoo dataset by exploiting the reduced dimensional space created by the *encoder* part of the CAE. We used a method of KDE that smooths over the distributions of samples to create a PDF so we could estimate outliers based on only a sample of the data. Although the model trained well and gave us some results of 4 outliers from a batch of 64, the outliers did not stand out as significant. we need more rigorous testing to allow us to understand the model further such as increased batch sizes, tests on known outliers and more analysis from multiple densities.

[1] en-USSDSS – Mapping the Universe ().

[2] D. Baron and D. Poznanski, enThe weirdest SDSS galaxies: results from an outlier detection algorithm, Monthly Notices of the Royal Astronomical Society **465**, 4530 (2017), arXiv:1611.07526 [astro-ph].

[3] L. Doorenbos, S. Cavuoti, M. Brescia, A. D'Isanto, and G. Longo, enComparison of outlier detection methods on astronomical image data (2021) pp. 197–223, arXiv:2006.08238 [astro-ph].

[4] Morphology/Classification | SDSS ().

[5] K. Sharma, A. Kembhavi, A. Kembhavi, T. Sivarani, and S. Abraham, enDetecting Outliers in SDSS using Convolutional Neural Network, Bulletin de la Société Royale des Sciences de Liège , 174 (2019).

[6] P. Wei, A. Luo, Y. Li, J. Pan, L. Tu, B. Jiang, X. Kong, Z. Shi, Z. Yi, F. Wang, J. Liu, and Y. Zhao, enMining unusual and rare stellar spectra from large spectroscopic survey data sets using the outlier-detection method, Monthly Notices of the Royal Astronomical Society **431**, 1800 (2013).

[7] Y. Yan, L. Cao, C. Kulhman, and E. Rundensteiner, enDistributed Local Outlier Detection in Big Data, in en*Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, Halifax NS Canada, 2017) pp. 1225–1234.

[8] D. Fustes, M. Manteiga, C. Dafonte, B. Arcay, A. Ulla, K. Smith, R. Borrachero, and R. Sordo, enAn approach to the analysis of SDSS spectroscopic outliers based on self-organizing maps: Designing the outlier analysis software package for the next *Gaia* survey, Astronomy & Astrophysics **559**, A7 (2013).

[9] B. Margalef-Bentabol, M. Huertas-Company, T. Charnock, C. Margalef-Bentabol, M. Bernardi, Y. Dubois, K. Storey-Fisher, and L. Zanisi, enDetecting outliers in astronomical images with deep generative networks, Monthly Notices of the Royal Astronomical Society **496**, 2346 (2020).

[10] D. Giles and L. Walkowicz, enSystematic serendipity: a test of unsupervised machine learning as a method for anomaly detection, Monthly Notices of the Royal Astronomical Society **484**, 834 (2019).

[11] S. Sarv Ahrabi, L. Piazzo, A. Momenzadeh, M. Scarpiniti, and E. Baccarelli, enExploiting probability density function of deep convolutional autoencoders' latent space for reliable COVID-19 detection on CT scans, The Journal of Supercomputing **78**, 12024 (2022).

[12] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE **86**, 2278 (1998), conference Name: Proceedings of the IEEE.

[13] D. Gries and O. Hazzan, enTexts in Computer Science, .

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (Massachusetts institute of Technology, 2016).

[15] D. R. Kübler, enIntroduction to Autoencoders (2022).

[16] K. W. Willett, C. J. Lintott, S. P. Bamford, K. L. Masters, B. D. Simmons, K. R. V. Casteels, E. M. Edmonson, L. F. Fortson, S. Kaviraj, W. C. Keel, T. Melvin, R. C. Nichol, M. J. Raddick, K. Schawinski, R. J. Simpson, R. A. Skibba, A. M. Smith, and D. Thomas, engGalaxy Zoo 2: Images from Original Sample (2013), type: dataset.

[17] E. Anello, enConvolutional Autoencoder in Pytorch on MNIST dataset (2022).

[18] D. S. Bhattiprolu, Python for Microscopists and other image processing enthusiasts (2022), original-date: 2019-06-10T17:53:14Z.

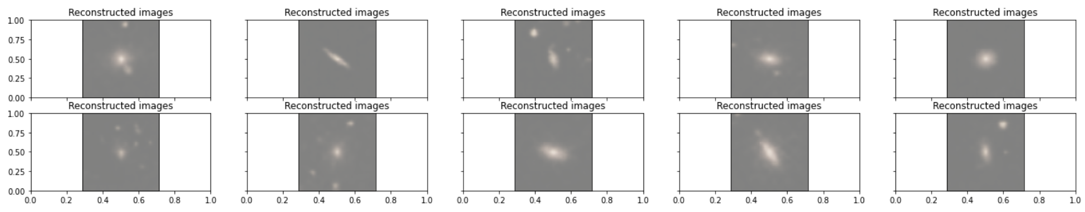

FIG. 6. Shows 10 original images before being placed through CAE.

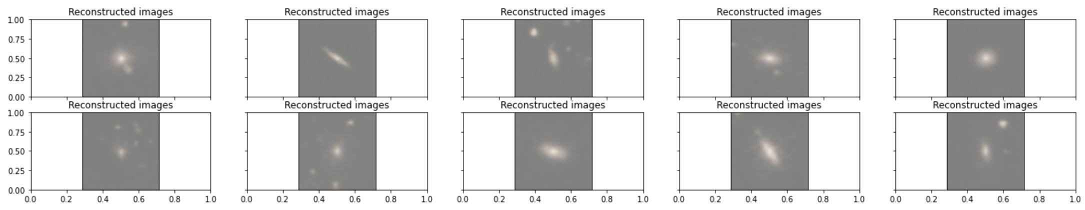

FIG. 7. Shows 10 Reconstructed images after being placed through CAE.

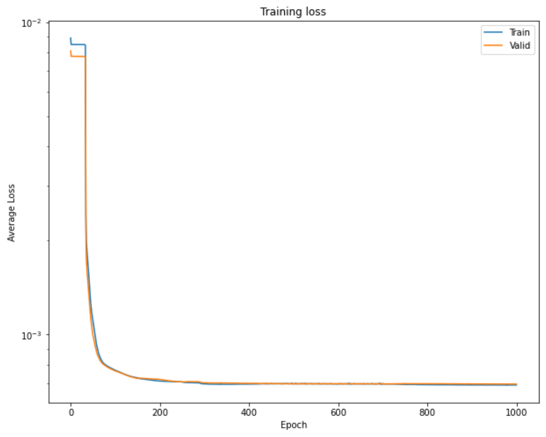

FIG. 8. Shows the CAE training and validation loss using *mean-squared error* .

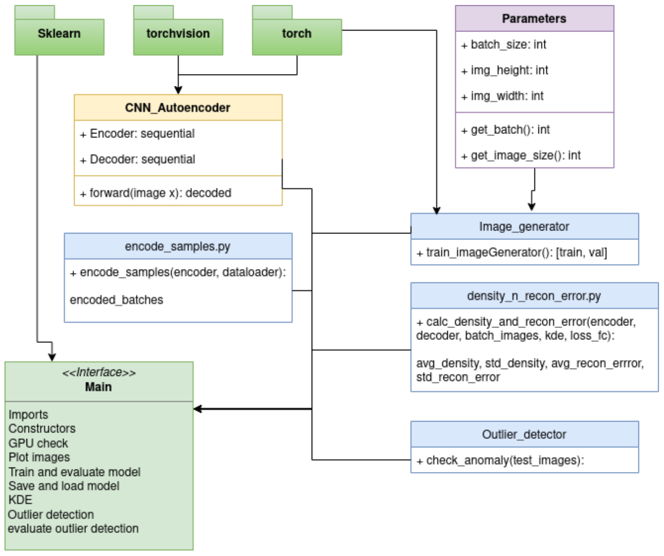

FIG. 9. Shows the relationship between class and functions. Main.py is the main interface where plots and functions will be run.