Literature Review – An exploration of unsupervised learning methods

Patel, J
Dept. of Physics and Astronomy
(University College London)
London, UK
11th November 2024

Scanning tunnelling microscopy (STM) is a powerful technique that can be used to generate high resolution images of surfaces at the atomic scale. Since its invention in 1982, STM has revolutionised the field of nanotechnology and has become an integral research tool found everywhere from semiconductor fabrication plants to various fields in material science [1, 2]. However, STM can very quickly generate vast amounts of complex image data that must be analysed by trained researchers, a labour-intensive and time-consuming task. The aim of this project is to develop a machine learning (ML) system that can detect and identify features and defects in STM images of silicon (001) surfaces.

Image recognition is a core task in ML, with applications ranging from identifying types of pastries or cancer cells, to algorithms for self-driving, or facial recognition in security systems [3-5]. It has become a standard task on which to benchmark ML algorithms. For this project, we want our system to be able to recognise the atomic grid of surface atoms, and subsequently identify defects such as atomic vacancies and step edges. This task is known as image segmentation.

When looking for patterns and relationships in data, dimensionality reduction is essential in ML, particularly for image-based datasets. The so-called "Curse of Dimensionality", coined by Bellman, refers to the exponential increase in data required to maintain reliable analysis as the number of dimensions grow [6]. For high dimensional data, datapoints are spread far apart, which makes it hard for ML models to find connections and relationships between datapoints, increasing computation time, and causing overfitting. To reduce the dimensionality of our image data, we will be using a type of neural network called an autoencoder.

Neural networks are a computational model inspired by the structure and function of biological neurons, designed to recognise patterns and make decisions. They consist of layers of interconnected neurons/nodes that process data through weights and activation functions, learning the complex relationships in data through iterative adjustments. For a feed-forward, "fully connected" network, we have three layers: input, hidden and output. The input layer receives the input data (images in our case) where each neuron corresponds to the dimensions of the input data (10,000 neurons for a 100x100-pixel image). Next, there is at least one hidden layer. Each neuron in a hidden layer takes an input from the previous layer, applies a set of weights and biases, and passes the result through an activation function. The output layer then outputs the final predictions of the model, depending on the task at hand. This type of "fully connected" network, as shown in fig.1, is not suitable for image classification as it lacks

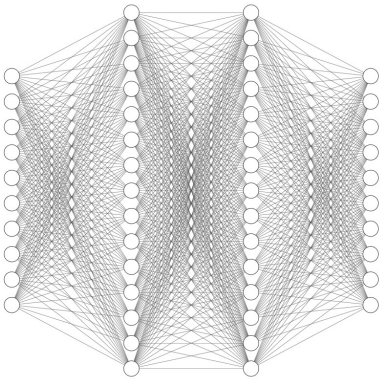

Figure 1 - Network architecture of a fully connected perceptron. One input layer, two hidden layers and an output layer. Generated using [7]

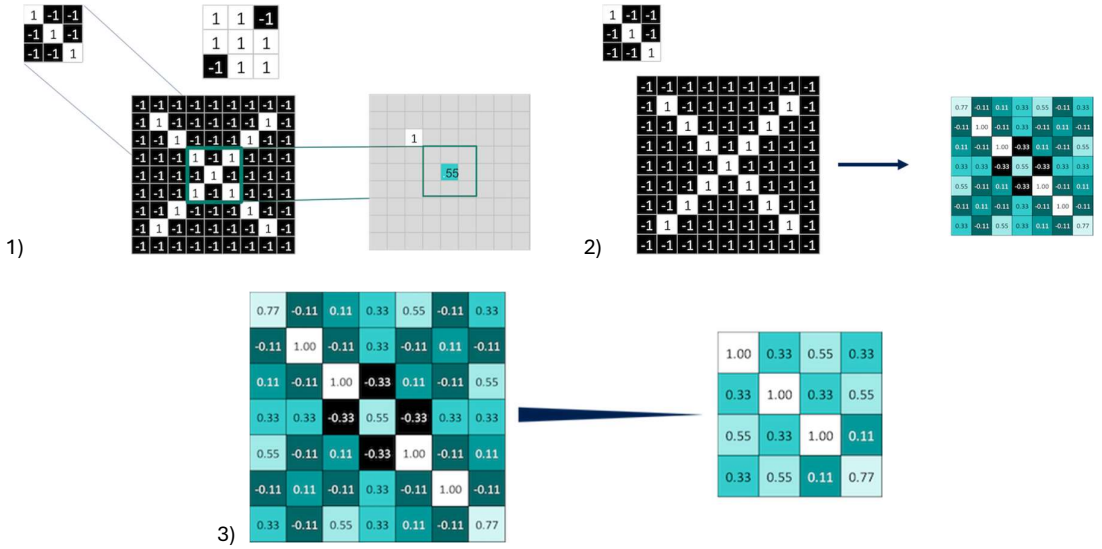

Figure 2 - A demonstration of convolution and pooling layers on an image. 1) Convolution kernel applied to an image. 2) The kernel's unique feature map. 3) A two-by-two tile pools the values across the feature map. Images from [8]

spatial invariance of features because the network processes each pixel individually. Instead, we use a convolution neural network as they better capture the spatial relationships between pixels.

Convolutional neural networks (CNNs) are far better for image recognition as they can "look" for local features, regardless of their position in image data (translational invariance). CNNs consist of alternating convolution and pooling layers to generate feature maps and reduce dimensionality. Convolution layers apply a kernel to an image, slide it across the image and learn specific features in it in various locations (fig. 2). Pooling layers reduce the dimensionality of the feature map by tiling it and taking the maximum value of each tile (fig. 2). Stacking convolution and pooling layers enable a CNN to extract features such as edges or textures in early layers, and shapes and objects in deeper layers. A typical CNN for classification ends with fully connected layers for producing class predictions.

Autoencoders (AEs) are a class of neural networks used for unsupervised learning tasks, designed to learn lower-dimensional representations of high-dimensional data like images. Unlike CNN classification networks that output a specific label, AEs focus on reconstructing the input data, learning compact representations through a bottleneck layer that captures essential features. CNN-based autoencoders are used for image data, where the convolutional layers help to capture spatial features. This makes them particularly useful for dimensionality reduction and image denoising.

Autoencoders consist of an encoder, a bottleneck, and a decoder (fig. 3). The encoder compresses the image into a more abstract form via convolution and pooling layers. The bottleneck consists of only a few neurons and forces the encoder to capture key features in a form called the latent vector. This latent vector is a compressed representation of the original image that contains core features that is optimised for reconstructing input images. From the bottleneck, the decoder reconstructs the original image from the latent vector via up-sampling and convolution layers. The AE is trained to minimise the difference between the input and output and

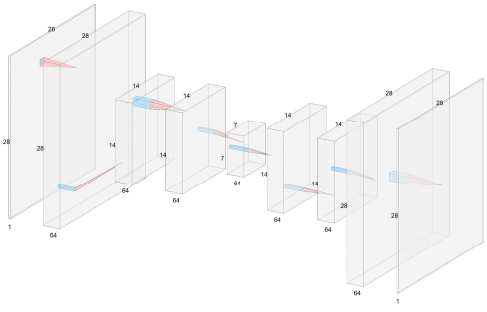

Figure 3 - An example of typical autoencoder architecture. This was used to generate latent vectors for the handwritten MNIST number dataset. Generated using [7]

make the decoder's output as close as possible to the input. Now that we have the images' much lower dimension latent vectors, we can more efficiently cluster them into groups.

Training a CNN from its original randomised weights generally requires an exceptionally large training set of labelled images. The ImageNet dataset for the ImageNet Large Scale Visual Recognition Challenge contained over 14,200,000 annotated images in its training set [9]. Supervised training methods require a label for each input image during model training and attempt to predict the label for previously unseen images [10]. For STM imaging, gathering and annotating data at this scale is extremely time consuming and labour intensive. Unsupervised learning is a branch of ML where users do not need to label samples or teach the model how to learn a mapping relationship. Instead, it allows the model to work on its own to discover patterns and learn information from the collection of unlabelled data. Clustering is the process of finding patterns in data to determine class labels without the use of labelled training data. It is useful when labelling data is costly or when class labels are difficult for a human to define.

Clustering

Most clustering algorithms fall into two basic types: partitioning and hierarchical. Partitioning methods construct K clusters where each group must contain at least one object, and each object must belong to exactly one group. Hierarchical methods, specifically agglomerative, start with all objects apart and then in each step clusters are merged until only one cluster is left. For either method, samples in the same cluster are similar to each other and dissimilar to samples in other clusters.

K-means clustering

K-means clustering is very commonly used and only requires prior knowledge of how many groups data should be grouped into. It partitions two-way, two-mode data (N images each with P variables describing the image features) into k classes ($C_1, C_2, \dots, C_k$). The algorithm aims to minimise the distance (sum of square error) between data points and their respective cluster centroids through the equation:

$$SSE(X, C_k) = \sum_{x \in X} \min_{c \in C_k} |x - c|^2$$

The algorithm starts by randomly selecting k cluster seeds and objects are assigned to the closest cluster. The centres of each cluster (centroid) are then recalculated, and objects are reassigned to the closest centroid. This is repeated until no more objects change cluster assignments [11].

K-means is commonly implemented because it can handle large image datasets efficiently, provides clear cluster boundaries, and works best when the data is expected to form roughly spherical clusters [12]. It has been used to help analyse imaging for breast cancer, to the automated analysis of TEM images of metal nanoparticles [13, 14]. In [13], after reducing the feature set via PCA, k-means clustering was used to classify data into groups, with a focus on identifying potential clusters of malignant and benign cases. This approach aided in the detection and categorisation of cancerous patterns resulting in a precision of 85.7% $\left(\frac{True\ Positive}{True\ Positive+Fa\quad Positive}\right)$ and a recall of 93.75% $\left(\frac{True\ Positive}{True\ Positive+Fa\quad Negative}\right)$. K-means' scalability and efficiency makes it suitable for large datasets, balancing computational demands with accuracy in feature extraction. In [14], k-means' centroid-based approach aligned well with the relatively spherical and homogeneous nature of metal nanoparticle images, as the clusters were approximately Gaussian-distributed in feature space.

An issue with k-means clustering is every time the algorithm is run, it falls into locally optimal solutions, and to find the global optimum it is often necessary to run the algorithm several times with different starting points and number of starting points. Choosing the optimal value of k can be

evaluated by techniques like the elbow method or silhouette analysis. The elbow method involves plotting the SSE as a function of the number of clusters and picking the elbow of the curve as the number of clusters to use. However, the elbow is not well-defined and is not reliable [15].

Another evaluation metric is silhouette analysis [16]. It is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette score ranges from −1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighbouring clusters.

DBSCAN

The DBSCAN algorithm, developed by Ester et al., is a density-based clustering algorithm noted for its ability to identify clusters of arbitrary shape, and ability to handle noise [17]. At its core, density-based algorithms cluster points based on regions of high density and define each cluster as areas of high density separated by areas of lower density. It requires two inputs: epsilon (ε), and the minimum number of points ($MinPts$). ε defines the radius around each point within which the algorithm searches for neighbouring points to assess density. $MinPts$ is the minimum required points within the ε-neighbourhood of a point to consider it as a core point.

To find a cluster, DBSCAN starts with an arbitrary point p and retrieves all points density-reachable from p. If p is a core point, this algorithm forms a cluster about p. If p is a border point, no points are density reachable, and the algorithm moves onto another point. Clusters are merged into one if they are close and if they fulfil the cluster requirements. A cluster is fully expanded once it reaches a point where no new points are density-reachable from the points in the cluster. DBSCAN then moves to the next unvisited point and repeats the process until all points are either assigned to a cluster or marked as noise.

The efficacy of DBSCAN is dependent on the operator and the understanding of the data. If the data and scale are not well understood, choosing a meaningful ε-radius can be difficult. The quality of the algorithm can also be affected by the distance metric used. It has recent revisions like OPTICS and HDDBSCAN that increase its robustness and speed [18, 19]. Density-based clustering methods are particularly useful in domains with irregularly shaped clusters and noise, like astronomy [20]. Its strength lies in its ability to identify noise, which helps in applications where outliers could signify anomalies, such as lesions in MRI scans.

The most similar works prior to ours are works by Ziatdinov et al. where they develop deep learning frameworks capable of identifying atomic species, defects, in STM images [21-23]. In [21], they used three separate CNNs to extract features from the STM dataset, which are then are clustered using mean-shift clustering – a type of density-based clustering similar to DBSCAN. In this study, for NN-1, simulated images were labelled with the positions of silicon dimer atoms and defects like

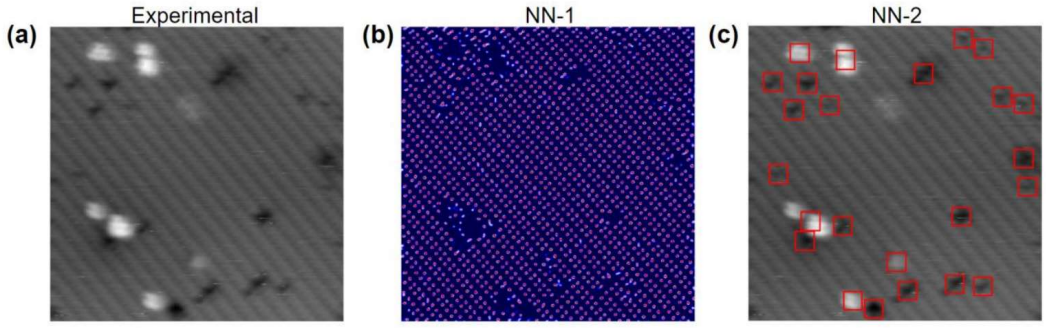

Figure 4 - An example of the application of a neural network for atom finding and defect finding. a) Example STM image data. b) All surface atoms found and located. c) Surface defects located and classified. Images from [21]

vacancies. For NN-2, experimental images were labelled with defect positions that disrupt the regular atomic structure. This labelling takes time, faces scalability issues, and limits the generalisation ability of the model and is where our approach differs.

Hierarchical Agglomerative Clustering

In hierarchical agglomerative clustering (HAC), the algorithm starts by treating each data point as its own cluster (leaves) and iteratively merges the most similar pairs of clusters into successively larger clusters (roots). The results can be visualised as a dendrogram, and the number of groups is chosen by "chopping" the tree at some point. There are two criteria that needs to be chosen by the operator: distance metric and linkage criterion. The choice of distance metric will influence the shape of the clusters, and the linkage criterion determines how distances between clusters are calculated, which in turn affects cluster assignments and the overall outcome of the process [24].

Hierarchical clustering is beneficial for analysing data with nested or hierarchical structures. In [25], it was used for cloud classification, allowing for multilevel analysis, and enabled the model to capture both large cloud structures and smaller internal features in a flexible, tree-like form. In [26] for 4D-STM analysis, hierarchical clustering aided in distinguishing structural differences and atomic-level variances without the need to specify the number of clusters in advance, which is advantageous when exploring complex material compositions that have diverse local structural patterns. HAC is formed "along the way" and it suffers from the problem that it can never "fix" what was done in previous steps.

Conclusions

In conclusion, machine learning techniques such as clustering and dimensionality reduction provide substantial benefits for automating the analysis of STM images. We believe that using autoencoders for dimensionality reduction combined with clustering methods can be effectively leveraged to identify atomic structures and defects in STM images of silicon surfaces. All three methods have their benefits and costs, but we will have to evaluate the performance of each when implementing them. We will build an autoencoder that reconstructs and extracts latent vectors of STM images and then evaluate the performance of different clustering techniques.

K-means could be advantageous for this task due to its computational efficiency and scalability, making it well-suited for large image datasets. In addition, its centroid-based approach is effective when clusters tend to be spherical, as demonstrated in applications like metal nanoparticle imaging. It does have limitations, and methods like the elbow method and silhouette analysis will be essential to evaluating the algorithm's performance.

HAC and DBSCAN offer further flexibility which may be useful for analysing STM images, where patterns may not follow simple shapes. HAC's key ability is producing a dendrogram that will enable multilevel analysis that can capture both coarse and fine features. Alternatively, DBSCAN provides advantages in handling noisy data or irregularly shaped clusters, which can be beneficial in certain STM image contexts where defects and features are not uniformly distributed.

In addition to clustering methods, preprocessing STM images using traditional image processing techniques to enhance edges and reduce noise should be explored. These steps are crucial for ensuring that the data fed into machine learning models is clean and relevant, which will ultimately improve the performance and accuracy of clustering algorithms. Overall, by integrating these preprocessing techniques and carefully selecting clustering methods, the automation of STM image analysis can be significantly improved.

References

[1] G. Binnig, H. Rohrer, Ch. Gerber, and E. Weibel, "Surface Studies by Scanning Tunneling Microscopy," *Physical Review Letters*, vol. 49, no. 1, pp. 57–61, Jul. 1982, doi: https://doi.org/10.1103/physrevlett.49.57.

[2] K. Nakamae, "Electron microscopy in semiconductor inspection," *Measurement Science and Technology*, vol. 32, no. 5, p. 052003, Mar. 2021, doi: https://doi.org/10.1088/1361-6501/abd96d.

[3] M. Turner, "BakeryScan and Cyto-AiSCAN," *Medium*, Nov. 13, 2021. https://towardsdatascience.com/bakeryscan-and-cyto-aiscan-52475b3cb779

[4] M. A. M. Elhassan *et al.*, "Real-time semantic segmentation for autonomous driving: A review of CNNs, Transformers, and Beyond," *Journal of King Saud University - Computer and Information Sciences*, p. 102226, Nov. 2024, doi: https://doi.org/10.1016/j.jksuci.2024.102226.

[5] L. Dai, Z. Luo, and S. Li, "Exploring Part Features for Unsupervised Visible-Infrared Person Re-Identification," *Proceedings of the 1st ICMR Workshop on Multimedia Object Re-Identification*, vol. 20, pp. 1–5, May 2024, doi: https://doi.org/10.1145/3643490.3661809.

[6] R. Bellman, *Adaptive control processes: a guided tour*. Princeton, N.J.: Princeton University Press, 1972. Available: https://www.jstor.org/stable/j.ctt183ph6v

[7] A. Lenail, "NN SVG," *alexlenail.me*. https://alexlenail.me/NN-SVG/index.html

[8] S. Wasswa, "How Convolutional Neural Networks Work.," *wasswa-sam.netlify.app*, Feb. 26, 2017. https://wasswa-sam.netlify.app/deeplearning/2017/02/26/how-convnets-work.html

[9] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, Apr. 2015, doi: https://doi.org/10.1007/s11263-015-0816-y.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2012, doi: https://doi.org/10.1145/3065386.

[11] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, Jun. 2010, doi: https://doi.org/10.1016/j.patrec.2009.09.011.

[12] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Englewood Cliffs, New Jersey: Prentice-Hall, 1988.

[13] A. Jamal, A. Handayani, A. A. Septiandri, E. Ripmiatin, and Y. Effendi, "Dimensionality Reduction using PCA and K-Means Clustering for Breast Cancer Prediction," *Lontar Komputer : Jurnal Ilmiah Teknologi Informasi*, p. 192, Dec. 2018, doi: https://doi.org/10.24843/lkjiti.2018.v09.i03.p08.

[14] X. Wang *et al.*, "AutoDetect-mNP: An Unsupervised Machine Learning Algorithm for Automated Analysis of Transmission Electron Microscope Images of Metal Nanoparticles," *JACS Au*, vol. 1, no. 3, pp. 316–327, Feb. 2021, doi: https://doi.org/10.1021/jacsau.0c00030.

[15] E. Schubert, "Stop using the elbow criterion for k-means and how to choose the number of clusters instead," vol. 25, no. 1, pp. 36–42, Jun. 2023, doi: https://doi.org/10.1145/3606274.3606278.

[16] P. J. Rousseeuw, "Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis," *Journal of Computational and Applied Mathematics*, vol. 20, no. 0377–0427, pp. 53–65, Nov. 1987, doi: https://doi.org/10.1016/0377-0427(87)90125-7.4

[17] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," 1996. Available: https://file.biolab.si/papers/1996-DBSCAN-KDD.pdf

[18] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "OPTICS," *ACM SIGMOD Record*, vol. 28, no. 2, pp. 49–60, Jun. 1999, doi: https://doi.org/10.1145/304181.304187.

[19] R. J. G. B. Campello, D. Moulavi, and J. Sander, "Density-Based Clustering Based on Hierarchical Density Estimates," *Advances in Knowledge Discovery and Data Mining*, vol. 7819, pp. 160–172, 2013, doi: https://doi.org/10.1007/978-3-642-37456-2_14.

[20] H. Yang, W. Ding, and C. Yin, "AAE-Dpeak-SC: A novel unsupervised clustering method for space target ISAR images based on adversarial autoencoder and density peak-spectral

clustering," *Advances in Space Research*, vol. 70, no. 5, pp. 1472–1495, Sep. 2022, doi: https://doi.org/10.1016/j.asr.2022.05.068.
[21] M. Ziatdinov, U. Fuchs, O. James, J. N. Randall, and S. V. Kalinin, "Robust multi-scale multi-feature deep learning for atomic and defect identification in Scanning Tunneling Microscopy on H-Si (100) 2x1 surface," *arXiv.org*, 2020. https://doi.org/10.48550/arXiv.2002.04716 (accessed Nov. 11, 2024).
[22] Maxim Ziatdinov *et al.*, "Deep Learning of Atomically Resolved Scanning Transmission Electron Microscopy Images: Chemical Identification and Tracking Local Transformations," *ACS Nano*, vol. 11, no. 12, pp. 12742–12752, Dec. 2017, doi: https://doi.org/10.1021/acsnano.7b07504.
[23] Maxim Ziatdinov *et al.*, "Building and exploring libraries of atomic defects in graphene: Scanning transmission electron and scanning tunneling microscopy study," *Science advances*, vol. 5, no. 9, Sep. 2019, doi: https://doi.org/10.1126/sciadv.aaw8989.
[24] F. Nielsen, *Introduction to HPC with MPI for Data Science*. Cham Springer International Publishing, 2016.
[25] T. Kurihana *et al.*, "Cloud Classification with Unsupervised Deep Learning," *arXiv.org*, 2022. https://doi.org/10.48550/arXiv.2209.15585 (accessed Nov. 11, 2024).
[26] K. Kimoto, J. Kikkawa, K. Harano, O. Cretu, Y. Shibazaki, and F. Uesugi, "Unsupervised machine learning combined with 4D scanning transmission electron microscopy for bimodal nanostructural analysis," *Scientific Reports*, vol. 14, no. 1, p. 2901, Feb. 2024, doi: https://doi.org/10.1038/s41598-024-53289-5.