Unsupervised Image Clustering Using Different CNN Architecture

S.Priyadharshini[1], D.R.Saranya[2]
[1, 2] Dept of CSE
[1, 2] Kings College of Engineering, Punalkulam , Near Thanjavur, Tamil Nadu , India

Abstract- *Clustering is a fundamental problem in many data-driven application domains, and clustering performance highly depends on the quality of data representation. Hence, linear or non-linear feature transformations have been extensively used to learn a better data representation for clustering. In recent years, a lot of works focused on using deep neural networks to learn a clustering-friendly representation, resulting in a significant increase of clustering performance. This project aims at providing insight on the transferability of deep CNN features to unsupervised problems. We study the impact of different pre trained CNN feature extractors on the problem of image set clustering for object classification as well as fine- grained classification. We propose a rather straightforward pipeline combining deep-feature extraction using a CNN pretrained on Image Net, VGG16 and Res Net and a classic clustering algorithm to clustering sets of images. This approach is compared to state-of-the-art algorithms in image-clustering and provides better results. These results strengthen the belief that supervised training of deep CNN on large datasets, with a large variability of classes, extracts better features than most carefully designed engineering approaches, even for unsupervised tasks.*

I. INTRODUCTION

Clustering is an interesting field of Unsupervised Machine learning where we classify datasets into set of similar groups. It is part of 'Unsupervised learning' meaning, where there is no prior training happening and the dataset will be unlabeled. Clustering can be done using different techniques like K-means clustering, Mean Shift clustering, DB Scan clustering, Hierarchical clustering etc. The key assumption behind all the clustering algorithms is that nearby points in the feature space, possess similar qualities and they can be clustered together.

In Introduction you can mention the introduction about your research

Up to know, we have only explored supervised Machine Learning algorithms and techniques to develop models where the data had labels previously known. In other words, our data had some target variables with specific values that we used to train our models. However, when dealing with real-world problems, most of the time, data will not come with predefined labels, so we will want to develop machine learning models that can classify correctly this data, by finding by themselves some commonality in the features, that will be used to predict the classes on new data.

Unsupervised learning main applications are:

- Segmenting datasets by some shared attributes.
- Detecting anomalies that do not fit to any group.
- Simplify datasets by aggregating variables with similar attributes.
- Dimensionality Reduction

Throughout this article we will focus on clustering problems and we will cover dimensionality reduction in future articles.

Clustering Analysis

In basic terms, the objective of clustering is to find different groups within the elements in the data. To do so, clustering algorithms find the structure in the data so that elements of the same cluster (or group) are more similar to each other than to those from different clusters.

In a visual way: Imagine that we have a dataset of movies and want to classify them. We have the following reviews of films:

The machine learning model will be able to infer that there are two different classes without knowing anything else from the data. These unsupervised learning algorithms have an incredible wide range of applications and are quite useful to solve real world problems such as anomaly detection, recommending systems, documents grouping, or finding customers with common interests based on their purchases.

Some of the most common clustering algorithms, and the ones that will be explored throughout the article, are:

- K-Means
- Hierarchical Clustering
- Density Based Scan Clustering (DBSCAN)
- Gaussian Clustering Model

K-Means Clustering

K-Means algorithms are extremely easy to implement and very efficient computationally speaking. Those are the main reasons that explain why they are so popular. But they are not very good to identify classes when dealing with in groups that do not have a spherical distribution shape.

The K-Means algorithms aims to find and group in classes the data points that have high similarity between them. In the terms of the algorithm, this similarity is understood as the opposite of the distance between datapoints. The closer the data points are, the more similar and more likely to belong to the same cluster they will be.

Squared Euclidean Distance

The most commonly used distance in K-Means is the squared Euclidean distance. An example of this distance between two points *x and y* in *m-dimensional* space

Here, *j* is the *jth* dimension (or feature column) of the sample points *x and y*.

- Adjusted Rand Index (ARI) € [-1,1]

To understand it we should first define its components:

- a: is the number of points that are in the same cluster both in C and in K
- b: is the number of points that are in the different cluster both in C and in K.
- n = is the total number of samples'
- The ARI can get values ranging from -1 to 1. The higher the value, the better it matches the original data.
- b = average distance to other sample i in closest neighboring cluster
- The Silhouette Coefficient (SC) can get values from -1 to 1. The higher the value, the better the K selected is. It penalized more if we surpass the ideal K than if we fall short. It is only suitable for certain algorithms such as K-Means and hierarchical clustering. It is not suitable to work with DBSCAN, we will use DBCV instead.

In Data Science, we can use clustering analysis to gain some valuable insights from our data by seeing what groups the data points fall into when we apply a clustering algorithm. Today, we're going to look at 5 popular clustering algorithms that data scientists need to know and their pros and cons!

II. MODULES

- Image Feature Extraction using ResNet and VGG16
- Data Preprocessing
- Extracting cluster-friendly deep features
- Using K Means to cluster a set of Images
- Network Updates

1.Image Feature Extraction

In this module pre-trained models can be used for image clustering. Initially we have implemented **ResNet 50** and **VGG 16** are a convolutional neural network model for image recognition proposed by different image dataset where **VGG16** refers to a VGG model with 16 weight layers.

VGG16: the input layer takes an image in the size of (224 x 224 x 3), and the output layer is a softmax prediction on 1000 classes. From the input layer to the last max pooling layer (labelled by 7 x 7 x 512) is regarded as the feature extraction part of the model, while the rest of the network is regarded as the classification part of the model.

2. Data Pre-processing.

In research, a large variety of datasets from multiplatform and heterogeneous sources need to be dealt with. Further, since clustering algorithms are used to discover hidden patterns from the data, CQ depends on the distributions of the data points and the underlying representation. Therefore, depending on problems and data types, different types of pre-processing may require depending upon CNN architectures

3. Extracting cluster-friendly deep features

In the context of clustering, after the training, the decoder part of an AE is no longer used but only the encoder is left, which acts as the feature extractor. LF then can be extracted from one or more layers (depending on the type of network architecture). For example, if extracted from a single layer, features come typically from the last layer of the network. However, if extracted from a multilayer or deep network (from any hidden layer or the deepest layer), it is found that LF can lead to better feature representations that

can enhance the separation of data points during the similarity computation.

4. Network Updates and Training

In this module, Training DL-based clustering algorithms may vary depending on the DNN architecture, different loss functions and training methods. However, since covering each of them in complete detail would be cumbersome in this comparative analysis, we discuss the detail of network updates and training for the pipeline methods (e.g. DEC) only that includes most of the possible steps explained in other DL-based approaches. In DL-based clustering, following two types of losses are optimized.

Non-clustering loss: this types of losses are independent of the clustering algorithm and usually enforces a desired constraint on the learned model, which guarantees that the learned representation preserves important information (e.g. spatial relationships between features) so the original input can be reconstructed in the decoding phase.
Clustering loss: this type of loss (e.g. RL1 and self-augmentation loss) is specific to the clustering method and the clustering-friendliness of the learned representations.

5.Using K means to cluster a set of Images

In this module, there are two important issues in applying k-Means in Scikit-Learn to this clustering problem. Num py. nd array. flatten to collapse a feature from the model (where different pre-trained models produce different shapes of features as listed below) into a one-dimension array required by k- Means in Scikit-Learn (where the input shape is *[n_samples, n_features]*).

Second, however, generally speaking, determining the number of clusters in a data set, or validating the assumption of our magic number, is a crucial step in solving a clustering problem. We will use silhouette coefficient to determine the number of clusters, and compare and contrast different pre-trained models later.

6.Evaluating the Performance of different clustering methods

In this module, we will consider two approaches to evaluate the performance of different clustering methods.

Internal Cluster Validation: investigating the structure of clustering results without information outside of the dataset, i.e., without the known labels. We will use Silhouette Coeffecient in Scikit-Learn for internal cluster validation. The measure is bounded between -1 for incorrect clustering and 1 for highly dense clustering. Scores around zero indicate overlapping clusters.

External Cluster Validation: comparing the results of a cluster method with the known labels. We will use Adjusted Rand Index in Scikit-Learn for external cluster validation. The range of score is between -1 and 1, where negative values mean that the predicted clusters and the known clusters are highly different, positive values mean that the predicted clusters and the known clusters are similar, and 1 is the perfect match score.We have compared different numbers of clusters (between 2 and 10) under 4 different pre-rained models (VGG16, VGG19, InceptionV3, and ResNet50). It is an interesting chart which explores many issues for discussion.

III. CONCLUSION

In this project, we provide a comprehensive review of DL-based clustering approaches for image datasets. Clustering results in three different use cases covering different types of data show that approaches based on DL outperform ML-based clustering algorithms. Firstly, transfer learning can be employed by means of pretrained models, e.g. ResNet50, Inception, ResNet, and VGG16/19 to extract deep features from the images. Weights of the first layers are kept intact, and only the last few layers are fine-tuned to get an improved feature representation. This could then be used to improve the DL- based clustering analysis by reducing the input dimensions to a lower number of features. However, since CNN-based pretrained models are trained on general- purpose images (e.g. ImageNet), they are often not suitable for imaging. Further experiments should show whether these improvements also carry over to DL- based clustering. And also we have learned how to build a keras model to perform clustering analysis with unlabelled datasets. Pre-trained auto encoder played a significant role in the dimensional reduction and parameter initialization, then custom built clustering layer was trained against a target distribution to refine the accuracy further.

REFERENCES

[1] T. Kohonen, ‘‘The self-organizing map,’’ Neurocomputing, vol. 21, nos. 1– 3, pp. 1–6, 1998.
[2] D. Reynolds, ‘‘Gaussian mixture models,’’ in Encyclopedia of Biometrics. Springer, 2015, pp. 827–832.
[3] M. Ester, H.-P.Kriegel, J. Sander, and X. Xu, ‘‘A density-based algorithm for discovering clusters in large spatial databases with noise,’’ in Proc. KDD, 1996, pp. 226–231.
[4] D. Arthur and S. Vassilvitskii, ‘‘k-means++: The advantages of careful seeding,’’ in Proc. 18th Annu.

ACM-SIAM Symp. Discrete Algorithms, 2007, pp. 1027–1035.

[5] J. A. Hartigan and M. A. Wong, ‘‘Algorithm AS 136: A k-means clustering algorithm,’’ J. Roy. Stat. Soc. C, Appl. Stat., vol. 28, no. 1, pp. 100–108, 1979.

[6] S. Wold, K. Esbensen, and P. Geladi, ‘‘Principal component analysis,’’ ChemometricsIntell. Lab. Syst., vol. 2, nos. 1–3, pp. 37–52, 1987.

[7] T. Hofmann, B. Schölkopf, and A. J. Smola, ‘‘Kernel methods in machine learning,’’ Ann. Stat., vol. 36, no. 3, pp. 1171–1220, 2008.

[8] A. Y. Ng, M. I. Jordan, and Y. Weiss, ‘‘On spectral clustering: Analysis and an algorithm,’’ in Proc. Adv. Neural Inf. Process. Syst., 2002, pp. 849–856.

[9] J. Schmidhuber, ‘‘Deep learning in neural networks: An overview,’’ Neural Netw., vol. 61, pp. 85–117, Jan. 2015.

[10] J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe, ‘‘Deep clustering: Discriminative embeddings for segmentation and separation,’’ in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Mar. 2016, pp. 31–35.

[11] J. Xie, R. Girshick, and A. Farhadi, ‘‘Unsupervised deep embedding for clustering analysis,’’ in Proc. Int. Conf. Mach. Learn., 2016, pp. 478–487.

[12] F. Li, H. Qiao, B. Zhang, and X. Xi. (2017). ‘‘Discriminatively boosted image clustering with fully convolutional auto-encoders.’’ [Online]. [13] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong. (2016). ‘‘Towards K-means- friendly spaces: Simultaneous deep learning and clustering.’’ [Online].

[13] K. G. Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang, ‘‘Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization,’’ in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 5747–5756.

[14] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou. (2016). ‘‘Variational deep embedding: An unsupervised and generative approach to clustering.’’ [Online].

[15] J. Yang, D. Parikh, and D. Batra, ‘‘Joint unsupervised learning of deep representations and image clusters,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2016, pp. 5147–5156.

[16] C.-C. Hsu and C.-W. Lin, ‘‘CNN-based joint clustering and representation learning with feature drift compensation for large-scale image data,’’ IEEE Trans. Multimedia, vol. 20, no. 2, pp. 421–429, Feb. 2018.

[17] Z. Wang, S. Chang, J. Zhou, M. Wang, and T. S. Huang, ‘‘Learning a taskspecific deep architecture for clustering,’’ in Proc. SIAM Int. Conf. Data Mining, 2016, pp. 369–377.

[18] E. Aljalbout, V. Golkov, Y. Siddiqui, and D. Cremers. (2018). ‘‘Clustering with deep learning: Taxonomy and new methods.’’ [Online]. [20] A. Ng, ‘‘Sparse autoencoder,’’ CS294A Lecture Notes, 2011.

[19] M. D. Boomija and M. Phil, ‘‘Comparison of partition based clustering algorithms,’’ J. Comput. Appl., vol. 1, no. 4, pp. 18–21, 2008.

[20] H.-P. Kriegel, P. Kröger, J. Sander, and A. Zimek, ‘‘Density-based clustering,’’ Wiley Interdiscipl. Rev., Data Mining Knowl. Discovery, vol. 1, no. 3, pp. 231–240, 2011.

[21] S. C. Johnson, ‘‘Hierarchical clustering schemes,’’ Psychometrika, vol. 32, no. 3, pp. 241–254, 1967.

[22] Goodfellow et al., ‘‘Generative adversarial nets,’’ in Proc. Adv. Neural Inf. Process. Syst., 2014, pp. 2672–2680.

[23] D. P. Kingma and M. Welling. (2013). ‘‘Auto-encoding variational Bayes.’’[Online].

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ‘‘ImageNet classification with deep convolutional neural networks,’’ in Proc. Adv. Neural Inf. Process. Syst., 2012, pp. 1097–1105.

[25] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, ‘‘Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations,’’ in Proc. 26th Annu. Int. Conf. Mach. Learn., 2009, pp. 609–616.

[26] G. E. Hinton, ‘‘A practical guide to training restricted Boltzmann machines,’’ Momentum, vol. 9, no. 1, pp. 599–619, 2010.

[27] D. Beeferman and A. Berger, ‘‘Agglomerative clustering of a search engine query log,’’ in Proc. 6th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2000, pp. 407–416.

[28] G. Chen. (2015). ‘‘Deep learning with nonparametric clustering.’’[Online].

[29] P. Huang, Y. Huang, W. Wang, and L. Wang, ‘‘Deep embedding network for clustering,’’ in Proc. 22nd Int. Conf. Pattern Recognit. (ICPR), Aug. 2014, pp. 1532–1537.

[30] X. Peng, J. Feng, S. Xiao, J. Lu, Z. Yi, and S. Yan. (2017). ‘‘Deep sparse subspace clustering.’’[Online]. Available: https://arxiv.org/abs/ 1709.08374

[31] H. W. Kuhn, ‘‘The Hungarian method for the assignment problem,’’ Nav. Res. Logistics, vol. 52, no. 1, pp. 7–21, 2005.