Deep evidential clustering of images

Loïc Guiziou[1], Emmanuel Ramasso[1], Sébastien Thibaud[1] and Sébastien Denneulin[2]

[1] Department of Applied Mechanics, FEMTO-ST Institute – UMR CNRS 6174 – ENSMM/UFC/UTBM, 25000 Besançon, France, emmanuel.ramasso@femto-st.fr
[2] SAFRAN CERAMICS, 33185 Le Haillan, France
sebastien.denneulin@safrangroup.com

Abstract. This paper presents a new image clustering method (DEEM) based on convolutional neural networks and the theory of belief functions used to encode uncertainty between clusters. The algorithm learns to generate mass functions for a given image through a training process that minimises a loss between the conflict computed from pairs of images and their dissimilarities. DEEM extends NN-EVCLUS and provides a gateway to the entire realm of deep learning, capitalising on all its advancements. It enables the full exploitation of the benefits offered by customisable layers, sophisticated optimisation algorithms, and other state-of-the-art techniques. DEEM can learn from the data itself, without requiring external labels but we can incorporate prior on labels if available as proposed in NN-EVCLUS. The first results are shown on the MNIST dataset (digit recognition).

Keywords: Belief functions · Unsupervised learning · Semi-supervised learning · Image clustering

1 Introduction

Clustering is a data analysis technique wherein data points are grouped based on their inherent similarities. The groups or clusters, are formed such that data points within the same cluster are more closely related to each other than to those in other clusters. Clusters can then be used to get insights in the data for further classification or interpretation.

Managing uncertainty in clustering is an important topic because real-world data often contains noise, outliers, and ambiguities, leading to uncertainty in the assignment of data points to clusters. By addressing uncertainty, clustering algorithms can produce more reliable and interpretable results, ensuring that the identified clusters accurately represent the underlying structure of the data.

In order to represent uncertainty, various formalisms can be used such as fuzzy sets [3, 18], probability theory [7, 15] or belief functions [4, 13]. Using belief functions, doubt between clusters is explicitly represented. In practice, doubt allows the end-user to visualise the contours of clusters which helps in making informed decisions and drawing meaningful insights from the clustering process.

When it comes to image clustering, there are two primary approaches. On one hand, features can be extracted from the input, such as color histograms or merged pixel blobs [10, 22]. On the other hand, the image itself can be treated as an input vector for conventional clustering methods. Despite the latter not fully considering the image structure, they can still yield satisfactory results. Both input methods may involve the use of deep clustering methods [17]. However, convolutional neural networks (CNN) are a class of method that leverages data structure for clustering. While often employed in supervised settings, recent applications of CNN have explored unsupervised and self-supervised learning. One common approach consists in a CNN serving as an encoder for the data, producing an output suitable for clustering methods like k-means. In the self-supervised setting, the clusters are used as a feedback to train the encoder like in [6]. The head of the CNN is then coupled with a MLP trained with the pseudo-labels using a discriminative loss.

Our approach, DEEM, diverges from traditional encoders as it directly provides cluster membership from input images, where the uncertainty is encoded by belief functions. Moreover, the option of partially or weakly supervised training is inherently integrated into the loss function, as detailed in the following section. DEEM is based on a recent method called NN-EVCLUS proposed by T. Denoeux [8]. Initially developed for clustering feature vectors with shallow networks, NN-EVCLUS is feature dependent, which requires to pay particular attention to feature relevance. DEEM handles image inputs which outstrips feature limitations. In addition, images can be processed by particular networks such as CNN, which have already shown great performance in supervised learning [12]. To our knowledge, this is the first neural network-based clustering method able to generate belief functions from images. By fully exploiting the benefits offered by customisable layers, sophisticated optimisation algorithms, and other state-of-the-art techniques, DEEM can represent a valuable method for image clustering under uncertainty and be used as a mass function generator.

The presentation of the method is described in Section 2 and Section 3 presents the results.

2 Method

2.1 Background on belief functions

A mass function on a finite set Ω is defined as a mapping of each element of the power set 2^Ω onto $[0, 1]$:

$$\begin{aligned} m : 2^\Omega &\mapsto [0, 1] \\ A &\to m^\Omega(A) \text{ s.c. } \textstyle\sum_A m^\Omega(A) = 1, m^\Omega(A) \geq 0 \end{aligned} \tag{1}$$

When $m^\Omega(S) > 0$, A is called a *focal set*, and if $m^\Omega(\emptyset) = 0$ then the basic belief assignment (BBA) is said *normal*. A mass function can be transformed into several other functions which allows the end-user to get insights about the

content of a mass and to make easier some computations [20, 21]. One of these functions is the plausibility defined as

$$Pl(A) = \sum_{B \cap A \neq \emptyset} m(B), \forall B \subseteq \Omega \tag{2}$$

As a particular case, the *contour function* related to m is defined as $pl : \Omega \mapsto [0, 1]$. It maps each singleton $\omega \in \Omega$ to its plausibility: $pl(\omega) = Pl(\{\omega\})$.

Given two mass functions, m_1 and m_2, defined on the same frame of discernment Ω, their combination through the conjunctive rule is

$$(m_1 \cap m_2)(C) = \sum_{A \cap B = C} m_1(A) m_2(B), \forall C \subseteq \Omega \tag{3}$$

From this combination, a conflict can arise when the intersection is empty between A and B:

$$\kappa = (m_1 \cap m_2)(\emptyset) = \sum_{A \cap B = \emptyset} m_1(A) m_2(B) \tag{4}$$

This conflict has been used in several algorithms based on belief functions such as target association [1] or the evidential hidden Markov model [16] since it is related to the likelihood. It has a very important role in NN-EVCLUS as shown subsequently.

2.2 NN-EVCLUS for feature vector clustering

NN-EVCLUS relies on a shallow neural network parameterised by $\boldsymbol{\theta}$ taking as inputs a d-dimensional feature vector $\boldsymbol{x}_i \in \Re^d$ and generating a vector of masses $\boldsymbol{m}_i$. Together, the BBA form a collection called credal partition $m = \{m_1, ..., m_N\}$ for N objects [9]. The number of outputs in the last layer of the network depends on the complexity of the mass that the end-user considered as necessary to represent the uncertainty on clusters. Classically, the singletons, the pairs and the whole frame are often sufficient. The remaining description does not depend on this limitation but due to the calculation complexity, adding more subsets will considerably increase the computation time.

The core idea of NN-EVCLUS is to quantify the conflict between each pair of mass functions $(\boldsymbol{m}_i, \boldsymbol{m}_j)$ and to use this conflict to update the network parameters. Therefore, NN-EVCLUS has some analogy with the siamese network (SN) [5] since the network parameters are fixed for a given pair of inputs $(\boldsymbol{x}_i, \boldsymbol{x}_j)$. In a SN, the true labels of individual inputs are not known, but we have to know whether each pair represents a similar (e.g. label "1") or a dissimilar (label "0") object. Based on this prior knowledge, the parameters of a SN are updated according to the cross-entropy loss or a contractive loss.

In NN-EVCLUS, the loss function is slightly different and the method unsupervised. It is defined as

$$\mathcal{L} = \sum_{i=1}^{N} \sum_{j>i} \left(\delta_{ij} - \kappa_{ij}(\boldsymbol{\theta})\right)^2 \tag{5}$$

where $\kappa_{ij}(\boldsymbol{\theta})$ is the conflict between the mass functions $\boldsymbol{m}_i$ and $\boldsymbol{m}_j$ generated after the forward pass using $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ as inputs and with network parameters $\boldsymbol{\theta}$. The value δ_{ij} is the dissimilarity between the two inputs. Dissimilarity can be computed in many ways. In the simplest case, we will use the Euclidean distance. The values are then scaled between 0 and 1 to be comparable to conflict. This approach presupposes that the greater the distance between images, the more likely the masses are to be in conflict. The main idea behind this loss is that if the network generates two conflicting masses while the inputs are similar, then the parameters should be updated in consequence. Conversely, if two inputs are similar and generate two masses with limited conflict, then the network behaves properly.

2.3 DEEM for image clustering

An implementation of the shallow version of NN-EVCLUS was done in R by T. Denoeux [8]. It requires inputs as feature vectors as defined in the original version. The implementation includes the possibility to tune up to three layers with an arbitrary number of neurons and using the ReLU activation function in all layers. Our first implementation was done in Matlab, with the possibility to manage any number of layers, to use all possible activation functions and various optimisers.

We then explored the possibility to improve this implementation in order to manage images as inputs and to change the loss. Therefore, the shallow network was replaced by a CNN. These networks, trained in an unsupervised manner, can extract useful visual features and representations that can be used for other tasks [6]. Implementation was done in both Matlab and Pytorch with GPU compatibility. The main contribution is to give the possibility to perform image clustering in an end-to-end manner without the need of manual feature extraction. Here, the objective of the CNN is to output the evidential partition of the images and does not aim to transform the images into features as auto-encoders do. As with the previous method, the output masses enable clustering and the calculation of the loss function, which depends on the conflict.

Thus, the input data consists of vectorised and normalised raw images. Given the large amount of data typically involved in general image databases, the implementation relies on minibatches. These minibatches are then processed by the CNN, which outputs the masses of the power set, including singletons, pairs, the empty set, and the universal set. Given the complexity of the computation depending on the number of elements and the number of focal sets, it is difficult to add higher-order intersections to the calculation. For 10 classes, this results in an output of 57 masses. The conflict is then extracted from these masses and integrated into the loss function. In the DEEM code, it is possible to change the loss function easily, but the configuration used for the tests remains the same Mean Squared Error (MSE) loss as in Eq. 5. Conversely to the general approach for image clustering which is based on prediction of the cluster assignments used as pseudo-labels in a discriminative loss, DEEM relies on the MSE loss defined in NN-EVCLUS. The way the dissimilarity δ_{ij} is computed is critical, in particular

for image since they have a structure conversely to standard features. If natural images are considered, several dissimilarity measures proposed in the past can be used [11].

In the original publication, T. Denoeux provided the gradient of the loss with respect to all parameters. These gradients can be useful according to the way the optimisation is implemented. In our case, when considering images, the convolutional layers particularly make these computations and implementation more difficult. Therefore, the gradients were found by automatic differentiation (AD) [2] which allows the end-user to easily configure the network as desired for a given application. The associated optimiser is Adam, which, once again, can be easily swapped with others such as Stochastic Gradient Descent with Momentum (SGDM) or Root Mean Squared Propagation (RMSProp). The DEEM code also allows for real-time calculation of ARI and other evaluators. This calculation is performed by evaluating the predicted clusters against the expected ones at each iteration. In our case, clustering is done by assigning the class of the subgroup where the plausibility function is the strongest. Thus, the plausibility calculation is performed on the 10 sets, with the winning cluster being the one with the highest value. Finally, it is difficult to define the stopping criterion for training the network because the loss varies greatly, and a threshold condition would not be adequate. For now, the number of iterations is the stopping criterion for training.

3 Results

3.1 Clustering performances on digits

The network's efficiency was initially evaluated using the MNIST standard dataset, which is made of images of the ten digits. One of the primary challenges involved configuring the network architecture and find a set of hyperparameters. The convergence outcomes are significantly influenced by the selection of layers and optimisation rules applied to each layer. One effective network configuration is illustrated in Figure 1.

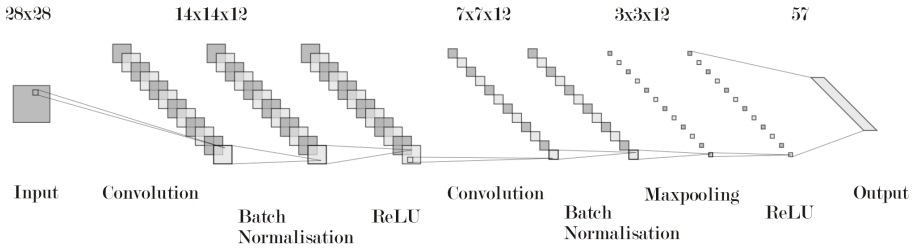


Fig. 1: Neural network for image handling of NNEVCLUS

Validation of the implementation – Initial testing was conducted under favorable conditions, recognising the substantial impact of dissimilarity calculations between images on convergence. Dissimilarities were first computed based on known labels, facilitating the network's convergence to an adjusted rand index (ARI) of 1 without any errors. Even though this method is not supervised in the sense that the labels are not directly incorporated into the loss function, it remains quasi-supervised. As a result, attempts to generalise on 60,000 images after training on 5,000 images achieve high scores: an ACC of 0.97, an NMI of 0.92, and an ARI of 0.93.

Grad-CAM analysis [19] on the test data revealed a logical distribution of areas of interest activated by neurons. Figure 2 depicts each class in a scenario where the network confused digits 6 and 8.

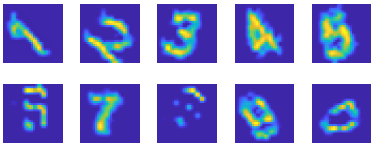


Fig. 2: Grad-CAM for each cluster

Using distance on images – When dissimilarities were calculated solely through cosine distance between images, without relying on labels, the network's performance decreased to an ARI of 0.25 and a rand index (RI) of 0.8. The disparity between ARI and RI arises from clustering multiple classes, thereby increasing the likelihood of errors across various classes. Errors predominantly occurred with digits that are challenging to distinguish or bear high resemblance, such as digits 1 and 7. This challenge is obvious in the dissimilarity matrix, where certain classes appear closely related. Besides, using distances between images to represent dissimilarity is a risky approach. Indeed, a simple translation between similar images significantly increases the distance. Therefore, other methods of computing dissimilarity can be employed. First, there is the extraction of image-specific features with methods such as Scale Invariant Feature Transform (SIFT) or Oriented FAST and Rotated BRIEF (ORB). Then, there are methods for comparing colour histograms, which are not suitable in our case where the images are black and white. Finally, there are methods that consider the structure of the data, such as Structural Similarity Index Measure (SSIM), or other contour detection methods like Fourier Descriptors, Shape Context Matching, and others. For now, this paper focuses solely on exploiting distances by applying dimensionality reduction to the images.

Using distance on reduced images – To enhance the reliability of the dissimilarity matrix, visualisation techniques and dimensional reduction algorithms such as principal components analysis (PCA) and uniform manifold approximation and projection (UMAP) were employed as it can improves clustering performance [14]. Utilising UMAP as an unsupervised method to project the data

and derive meaningful dissimilarities based on distance significantly improved clustering performance, resulting in an ARI increase up to 0.65. In addition, using t-distributed Stochastic Neighbor Embedding (t-SNE) reduction increases the ARI up to 0.73.

Using pre-clustering on reduced images – As we have seen, the closer the dissimilarity matrix is to the labels, the easier it is for the algorithm to converge. However, there's nothing to rule out the use of an initial unsupervised clustering to obtain the dissimilarity matrix. The interest of this method then lies in the generalisation of this model. Indeed, on 60,000 test images, a conventional method consisting in reducing the images via UMAP and then applying clustering with K-means is relatively time-consuming, taking 25 seconds on average compared with 0.5 for DEEM. In addition, DEEM's generalisability is much better, as shown in Table 1.

	DEEM	K-means	Hierarchical
ACC	**0.96**	0.84	0.83
NMI	**0.89**	0.75	0.76
ARI	**0.91**	0.71	0.76

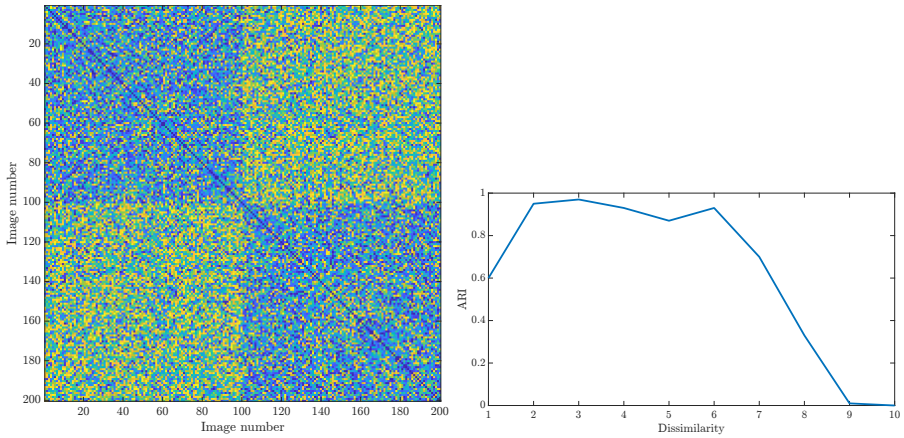
Table 1: Comparison of clustering methods on 60K testing MNIST images

3.2 Effect of dissimilarity calculation method

As specified in Section 2.2, the loss function calculates the difference between conflict and dissimilarity of two objects. For this reason, the calculation of the later is crucial for clustering. Different distances calculation methods can be employed such as Euclidean distance or cosine distance among others. Furthermore, extracted features as for original NN-EVCLUS can be used to generate the dissimilarity matrix instead of the set of images. To illustrate the dependence of the network on the dissimilarity, clustering was carried out on a two-class digit clustering for several dissimilarity matrices. To do so, a set of 10 matrices were designed from the perfect binary one to fuzzier ones.

As shown on Figure 3a, a random Gaussian noise is added to the previous binary matrix for different range of standard deviation ranging from 0 to 0.8. Each of the 10 clustering was performed 30 times to take into account variability in the convergence. Finally, results are expressed thanks to ARI. The mean of results for each matrix is shown on Figure 3b. The algorithm decently tolerates fuzziness until the seventh point corresponding to 0.53 standard deviation of the Gaussian noise added, which is pretty much encouraging. After that, the algorithm struggles to achieve convergence. Moreover, Gaussian noise was chosen to be the closest from real cases, but conventional random noise also provides similar results. Finally, the first point of the diagram draws our attention to its low value. Indeed, perfect dissimilarity leads to a drop in performance. In such a case, the loss function seems to be highly sensitive to small changes. This leads to instability during training, with small fluctuations in the predicted

dissimilarities causing disproportionately large changes. Using ADAM optimiser, a lower learning rate helps to get through the issue and gives a perfect percentage of convergence. The architecture was not optimised which can also explains the sensitivity.



(a) Dissimilarity matrix with added Gaussian noise

(b) ARI variation as a function of fuzziness in dissimilarity

Fig. 3: Noised distance experimentation

4 Conclusion

DEEM is an extended version of NN-EVCLUS algorithm to handle images instead of feature vectors. The layers and optimisers were adapted for a highly configurable neural network. We firstly showed performance of the new algorithm on MNIST data in a favorable configuration where distances where determined thanks to the labels. By the desire to have a true unsupervised learning, the next tests were conduced using distance dissimilarity which leads to confusion between close digits such as 1 and 7. In order to improve the performance, dissimilarity where calculated on UMAP reduction, and highly increased the ARI. Finally, further work remains to be done, particularly in try to cluster acoustic emission spectrograms. The results of DEEM were promising on several applications, either on standard benchmarks and from our laboratory. Its generalisation capabilities were also highly encouraging. Current work is on an extensive experimental validation of DEEM and its comparison with other algorithms of the literature on large datasets. Finally, as NN-EVCLUS did, it is possible to add constraints to some of the labels. This feature seems promising, but has yet to be quantified in terms of the number of apriori used.

Acknowledgments. This work has been achieved in the frame of the EIPHI Graduate school (contract "ANR-17-EURE-0002"), with the financial support of SAFRAN CERAMICS and Agence de l'innovation de défense (AID).

References

1. Ayoun, A., Smets, P.: Data association in multi-target detection using the transferable belief model. International Journal of Intelligent Systems **16**(10), 1167–1182. https://doi.org/10.1002/int.1054
2. Bartholomew-Biggs, M., Brown, S., Christianson, B., Dixon, L.: Automatic differentiation of algorithms. Journal of Computational and Applied Mathematics **124**(1), 171–190 (2000). https://doi.org/10.1016/S0377-0427(00)00422-2
3. Bello, R., Falcón, R., Pedrycz, W., Kacprzyk, J.: Granular Computing: At the Junction of Rough Sets and Fuzzy Sets. Springer Berlin Heidelberg, Berlin, Heidelberg (2008). https://doi.org/10.1007/978-3-540-76973-6
4. Ben Hariz, S., Elouedi, Z., Mellouli, K.: Clustering approach using belief function theory. Artificial Intelligence: Methodology, Systems, and Applications **4183**, 162–171. https://doi.org/10.1007/11861461_18
5. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a "siamese" time delay neural network. Advances in neural information processing systems **6** (1993). https://doi.org/10.1142/S0218001493000339
6. Caron, M., Bojanowski, P., Joulin, A., Douze, M.: Deep clustering for unsupervised learning of visual features. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision – ECCV 2018, vol. 11218, pp. 139–156. Springer International Publishing. https://doi.org/10.1007/978-3-030-01264-9_9
7. Chen, L., Brown, S.D.: Bayesian estimation of membership uncertainty in model-based clustering. Journal of Chemometrics **28**(5), 358–369. https://doi.org/10.1002/cem.2511
8. Denœux, T.: NN-EVCLUS: Neural network-based evidential clustering. Information Sciences **572**, 297–330. https://doi.org/10.1016/j.ins.2021.05.011
9. Denœux, T., Masson, M.H.: Evclus: Evidential clustering of proximity data. IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society **34**, 95–109 (2004). https://doi.org/10.1109/TSMCB.2002.806496
10. Gao, B., Liu, T.Y., Qin, T., Zheng, X., Cheng, Q.S., Ma, W.Y.: Web image clustering by consistent utilization of visual features and surrounding texts. In: Proceedings of the 13th annual ACM international conference on Multimedia. pp. 112–121. ACM. https://doi.org/10.1145/1101149.1101167
11. Goshtasby, A.A.: Similarity and dissimilarity measures. Image Registration pp. 7–66. https://doi.org/10.1007/978-1-4471-2458-0_2
12. Li, Z., Liu, F., Yang, W., Peng, S., Zhou, J.: A survey of convolutional neural networks: Analysis, applications, and prospects. IEEE Transactions on Neural Networks and Learning Systems **33**(12), 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827
13. Liu, Z.g., Pan, Q., Dezert, J., Mercier, G.: Credal c-means clustering method based on belief functions. Knowledge-Based Systems **74**, 119–132. https://doi.org/10.1016/j.knosys.2014.11.013
14. M, A.: Considerably improving clustering algorithms using umap dimensionality reduction technique: A comparative study. pp. 317–325. https://doi.org/10.1007/978-3-030-51935-3_34
15. Melnykov, V.: Challenges in model-based clustering. WIREs Computational Statistics **5**(2), 135–148. https://doi.org/10.1002/wics.1248

16. Ramasso, E.: Contribution of belief functions to hidden markov models with an application to fault diagnosis. In: 2009 IEEE International Workshop on Machine Learning for Signal Processing. pp. 1–6. IEEE (2009). https://doi.org/10.1109/MLSP.2009.5306209
17. Ren, Y., Pu, J., Yang, Z., Xu, J., Li, G., Pu, X., Yu, P.S., He, L.: Deep clustering: A comprehensive survey (2022)
18. Sadeghian, A., Mendel, J.M., Tahayori, H.: Advances in Type-2 Fuzzy Sets and Systems: Theory and Applications. Springer New York (2013). https://doi.org/10.1007/978-1-4614-6666-6
19. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-CAM: Visual explanations from deep networks via gradient-based localization. International Journal of Computer Vision **128**(2), 336–359. https://doi.org/10.1007/s11263-019-01228-7
20. Sentz, K., Ferson, S.: Combination of evidence in dempster-shafer theory (SAND2002-0835, 800792). https://doi.org/10.2172/800792
21. Smets, P., Kennes, R.: The transferable belief model. Artificial Intelligence **66**(2), 191–234 (1994). https://doi.org/10.1016/0004-3702(94)90026-4
22. Yee Leung, Jiang-She Zhang, Zong-Ben Xu: Clustering by scale-space filtering. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(12), 1396–1410. https://doi.org/10.1109/34.895974