Unsupervised learning using CNNs

Sara Raris Miralles

September 2019

1 Introduction

The goal of this report is to document all the steps performed for assessing unsupervised learning using CNNs on a given dataset. The first part is focused on learning and feature extraction. For this, several CNN architectures have been proposed and trained on the training set. Then, one of their layers have been selected for feature extraction. The second part is focused on clustering. In this part, the trained models are used for extracting features from the test set and a clustering method is applied to those features in order to obtain the clusters.

2 Dataset description

In the first place, it is important to define the dataset used in this task. This dataset is split into training and test set consisting on 2006 and 466 images respectively. The images constituting the dataset belong to three different classes.

One of the most important parts in the learning process is having an adequate dataset. This involves having a good distribution amongst the classes. In this particular case, the dataset is not balanced. The third class makes up for half of the dataset itself. Having such an imbalance can result on the model giving more importance to the class that has more examples when learning. Thus, prioritizing the biggest class and resulting on a poorer learning of the classes with less examples.

There are several ways of dealing with this issue. The first one would be to collect more data, which is impossible in this case. The second one, would be to augment the available data in order to have more examples in the smaller classes. However, since the images were not conventional transforming them could have been tricky and this option was discarded as well. Finally, the chosen option is giving more importance to the less represented classes during the training. This can be done using *class weight* option in Keras.

3 Learning phase

3.1 Training

In order to extract relevant features for clustering them, a model has to be trained first. For this task, several models were trained on the train set. The dataset has about enough examples for training a model from scratch. However, the amount of data available is not very large and training from scratch is very time consuming. To solve this issue, the models proposed have been previously trained on ImageNet.

The models that have been tested have been selected from Keras. Since Keras does not allow ImageNet weights to be loaded when the number of classes is modified, the last *Dense* layer with 3 neurons has been added afterwards. The tested models are the following ones:

1. ResNet50: trained for 15 epochs with a batch size of 56.
2. InceptionResNetV2: trained for 25 epochs with a batch size of 56.
3. Xception: trained for 25 epochs with a batch size of 56.

3.1.1 Models with non-clustering loss

The first approach selected is the simpler one. In this first approach, the models listed above have been trained using a non-clustering loss, cross-entropy loss. This is because even if a neural network has a non-clustering loss, the features extracted can be used for clustering after training. The neural network is used in this case for changing the representation of the input. Even though this transformation is beneficial sometimes, this approach does not guarantee good results. Moreover, the optimizer used in these models is stochastic gradient descent. Results of the training can be found in section 5.1.1.

3.1.2 Models with clustering loss

Using a clustering loss when training the neural network has the potential of yielding features that are more clustering friendly. As the results in section 5.3 show, cross-entropy loss does not produce features that are easy to cluster. In an attempt to improve this results, training with clustering loss has been explored. To do this, the ResNet50 previously trained has been retrained with mean absolute error (MAE) loss.

In the first place, the centers of the clusters have been computed. In order to to this, the classification layer of the trained ResNet50 has been removed. Then the features from all the images in the training set have been extracted and the mean has been computed. This has produced three different cluster centers, Figure 1. Once the centers have been obtained, ResNet50 has been trained again removing the last layer. The mean absolute error between the output features of each of the training images and their corresponding cluster center has been minimized during the training. This model has been trained using stochastic gradient descent, batch size of 32 and during 35 epochs. Results can be seen in section 5.1.2.

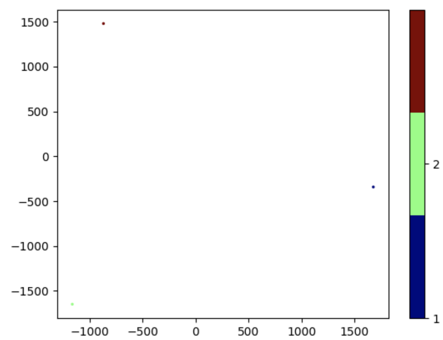


Figure 1: Mean of all the features of the training set images extracted with trained ResNet50.

3.2 Feature extraction

After the models have been trained, they are used for extracting features on the test set. In order to do this a specific layer needs to be selected. As the semantic information needed for distinguishing between classes is found in the final layers of the networks, the penultimate layer has been selected for feature extraction. This layer is found right before the final softmax classification layer and is an average pooling layer.

4 Clustering

Once features have been extracted from the test set using the models proposed, the clustering begins. The two clustering algorithms that show better results on the state of the art [2] have been selected: K-means clustering and aglomerative clustering. The results can be seen in section 5.2

The evaluation metric used for the clustering are normalized mutual information (NMI) and adjusted random index. While NMI score results are between 0 (no mutual information) and 1 (perfect correlation), ARI score results are between -1 and 1. When using ARI, muasurements close to 0.0 are considered random. These two metrics have been chosen as they are a good way of evaluating cluster accuracy.

5 Results

In this section the results for both the learning phase and the clustering are discussed.

5.1 Learning

It is important to evaluate the learning capability of the models proposed. In other words, see how good these neural networks learn the patterns of the dataset. However, achieving high accuracy in the training phase does always mean that the features will be adequate for performing unsupervised learning.

5.1.1 Models with non-clustering loss

The plots for the accuracy and the loss for the training of ResNet50, InceptionResNetV2 and Xception can be seen respectively in Figure 2, Figure 3 and Figure 4. In order to be able to spot overfitting during the training, these networks were validated on the test set. As it can be observed, all of the models obtained good results and overfitting was not an issue. ResNet50 converged faster than the other two networks taking less than 15 epochs to converge. Both Xception and InceptionResNetV2 took between 20 to 25 epochs.

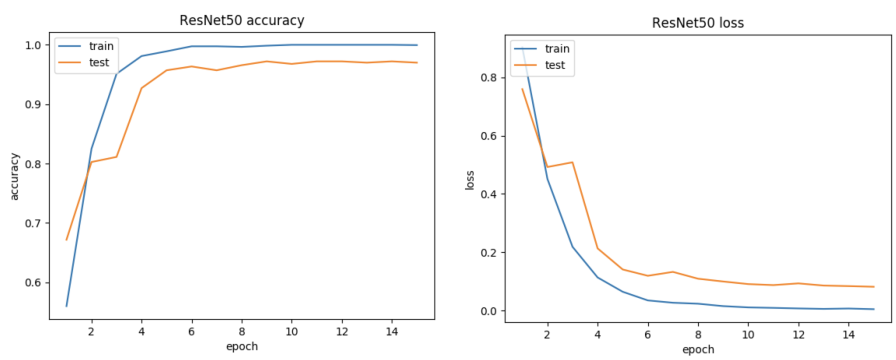


Figure 2: Train and test loss for ResNet50.

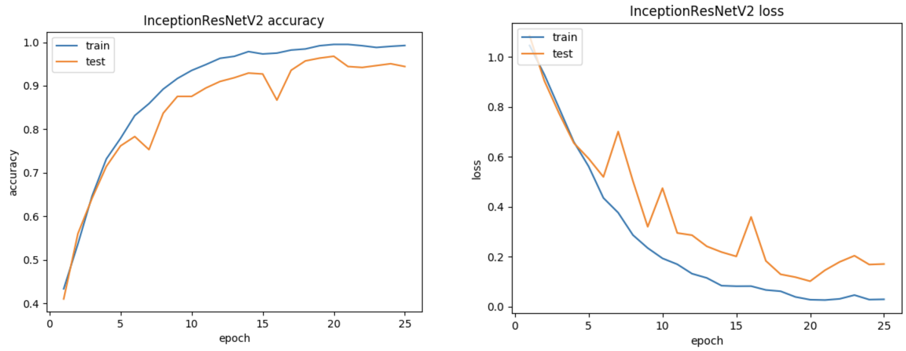


Figure 3: Train and test loss for InceptionResNetV2.

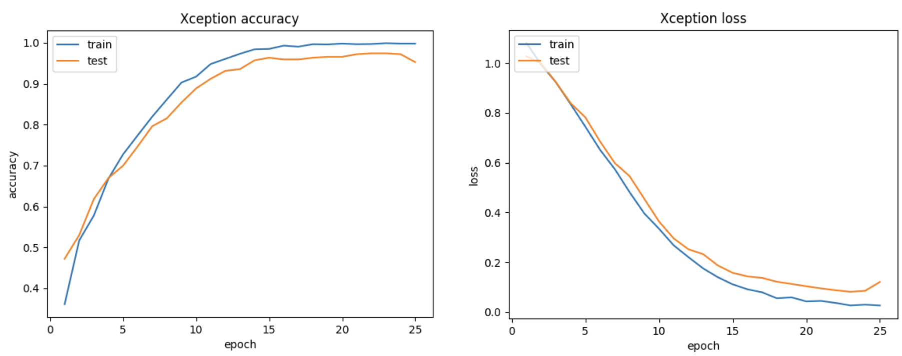


Figure 4: Train and test loss for Xception.

5.1.2 Model with clustering loss

The loss plot for ResNet50 trained with the clustering loss proposed can be seen in Figure 5. The loss is rather high since the features being compared have many dimensions. However, the loss steadily drops during the training.

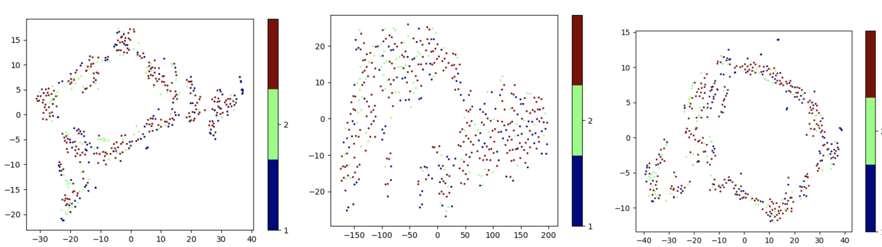
Figure 5: Train and test loss for ResNet50 trained with clustering loss.

5.2 Clustering

The features extracted from the test set have been clustered using K-means and aglomerative clustering. In order get an idea on how these features look, PCA has been applied to reduce dimensionality.

5.3 Models with non-clustering loss

Figure 6 shows the features extracted for each of the models proposed after reducing its dimensionality by applying PCA. As it can be observed, clusters can not be differentiated and all of the classes are mixed together. This is already a clear indicator that attempting to cluster these features will not provide good results.



(a) ResNet50 features (b) InceptionResNetV2 features (c) Xception features

Figure 6: 2D representation of the features extracted for the models.

Results of the evaluation of the proposed models can be seen in Table 1 and show that the clustering quality is not good. The model that achieves better performance is ResNet50 with Kmeans, with a NMI of **0.0397** and ARS of **0.0068**.

Model tested	NMI	ARS
ResNet50 Kmeans	0.0397	0.0068
ResNet50 AC	0.0253	0.0115
InceptionResNetv2 Kmeans	0.0103	-0.0021
InceptionResNetV2 AC	0.0109	-0.0175
Xception Kmeans	0.0117	-0.0021
Xception AC	0.0077	-0.0073

Table 1: Cluster accuracy for the three proposed models with both Kmeans clustering and Aglomerative Clustering.

5.3.1 Model with clustering loss

After training ResNet50 with clustering loss, features have been extracted for the test set. PCA has been applied to these features in order to represent them in a 2D space. However, as it can be seen in Figure 7, the clusters are still not visible and the features are still mixed together. This is further proven after applying both Kmeans and aglomerative clustering and evaluating the clusters with the chosen metrics, NMI and ARS. With a NMI of **0.01784** and an ARS of **0.00211**, cluster quality has not been enhanced by the chosen loss.

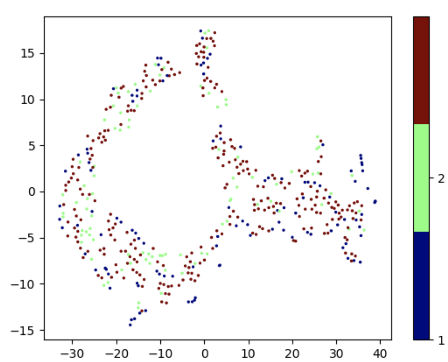


Figure 7: 2D representation of the features extracted for ResNet50 trained with clustering loss.

5.3.2 Feature analysis

After obtaining such results for both models with non-clustering loss and the model with cluster loss, the features extracted have been analyzed. By extracting the features from the test set images and computing the mean across all three classes it can be observed that the feature vectors are very similar. In fact, the normalized feature vectors obtained for each of the clusters are so similar that their differences can not be appreciated by looking at them. This happens both for the models trained with non-clustering loss and the model trained with clustering loss. Two examples can be observed, features extracted with InceptionResNetV2 in Figure 8 and features extracted with ResNet50 trained with clustering loss 9.

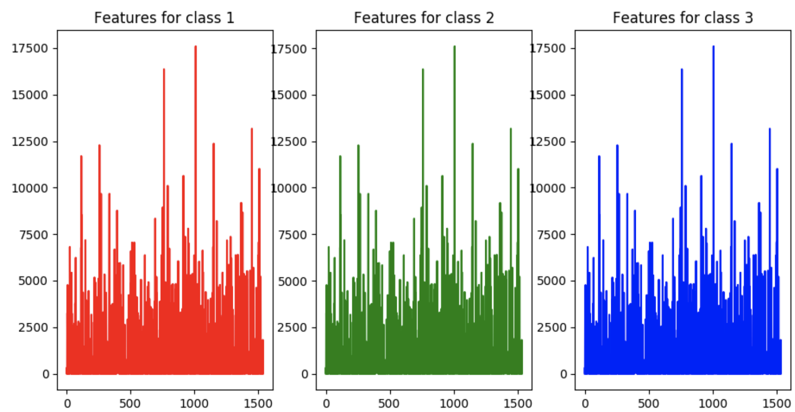


Figure 8: Normalized features for each of the images in the test set for the three classes extracted using InceptionResNetV2 with non-clustering loss.

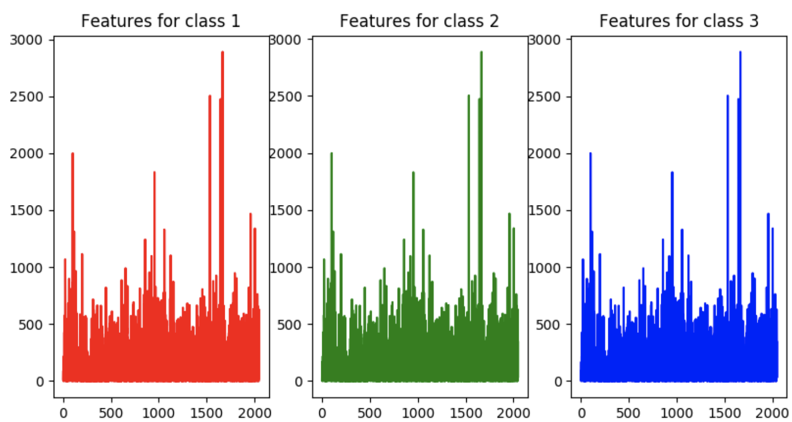


Figure 9: Normalized features for each of the images in the test set for the three classes extracted using ResNet50 with clustering loss.

6 Conclusion

This task has focused on performing unsupervised learning using deep learning. Models with both non-clustering and clustering loss have been proposed for this purpose. Although the models with non-clustering class have yielded good results during the training phase, the features that have produced are not adequate for performing clustering. Training a model with a clustering loss had the purpose of improving the clustering of the classes. However, this did not happen and the quality of the clustering was in fact worse than before.

After carefully comparing the features extracted for all the classes it is easy to see why it is so hard to group them into clusters. The features for all the classes are very similar to each other. The networks have learned how to differentiate between as they have been trained to do so with the labels. However, performing unsupervised learning on the features extracted is not an easy task.

Trying another type of neural network, such an encoder-decoder [1], or another loss, such as one that focuses on image similarities [3], could help with this issue. However, this might be trickier as the images are very similiar to each other.

References

[1] Jianlong Chang et al. "Deep Adaptive Image Clustering". In: *Proceedings of the IEEE International Conference on Computer Vision* 2017-October (2017), pp. 5880–5888. ISSN: 15505499. DOI: `10.1109/ICCV.2017.626`.

[2] Joris Guérin et al. "CNN Features are also Great at Unsupervised Classification". In: (2018), pp. 83–95. DOI: `10.5121/csit.2018.80308`. arXiv: `arXiv:1707.01700v2`.

[3] Lustering With. "Under review as a conference paper at ICLR 2018 CLUSTERING WITH DEEP LEARNING: TAXONOMY AND NEW METHODS". In: (2018), pp. 1–12. URL: `https://openreview.net/pdf?id=B1eT9VMgOX`.