Low-Shot-Learning of Diseases in Chest X-Rays via Hallucination

Matan Harel Uri Avron Jonathan Somer *

September 4, 2018

Abstract

One of the promises of the recent advancements in Artificial Intelligence is the ability to facilitate high precision computer aided diagnosis (CAD) systems and make such high precision diagnosis affordable and highly available. Medical imaging is one area where current deep learning and computer vision methods could possibly be applied. These methods utilize large sets of labelled images in order to achieve high accuracy. High quality labelled data is difficult to obtain due to a variety of factors.

Our objective is to answer whether we can use an existing large set of X-Ray images of healthy patients as well as a large of set of images of patients with some diseases, in order to improve the learning accuracy of new diseases for which we have only a few example images to learn from. This setting is known as Low-Shot-Learning.

We have implemented, extended and re-evaluated a method for performing low-shot-learning proposed by Bharath Hariharan and Ross Girshick from Facebook AI research, 2016 [1]. During this process we have also attempted to answer some questions that remained open while reading their innovative original paper and we show some novel methods for evaluating this low-shot-learning setting.

1. Introduction

1.1. Low-Shot-Learning

The ability to learn from very few examples is a hallmark of human visual intelligence. **Classical Machine Learning approaches fail to generalize[add reference]** from few examples so new techniques are required for performing Low-Shot-Learning.

The setting for low-shot-learning is composed of two phases. The first is a representation learning phase: the learner tunes its feature representation on an available set of base classes that have many training instances. In the low-shot learning phase, the learner is exposed to a set of novel classes with only a few examples per class and must learn a classifier over the joint label space of base and novel classes.

We evaluate the new classifier's accuracy over both the base and novel classes in order to see that higher accuracy was achieved on the novel classes but also that accuracy was not impaired for the base classes.

Low-Shot-Learning is a great challenge particularly in our setting as learning in a medical setting carries many unique challenges. Each disease category contains great intra-class variation: patients have varying anatomies, there are different methods of performing each examination, variation might be induced by different equipment and so on. Thus, very large labelled datasets are needed in order to capture this great variation. Obtaining high quality labelled datasets as such is very difficult. In addition, even if labelled data is obtained, the physician's diagnosis can be incorrect and in many cases is not validated or such validation does not get logged. In the dataset we chose to work with the researches who gathered the data employed a NLP text mining solution to procure labels from physicians written reports, a process which **possibly adds further noise[how much exactly + ref.]** to the labelling.

Thus, high quality labelled data is difficult to obtain and the ability to learn from little data is highly valuable.

1.2. Our Approach - High Level Glance

As noted, our approach is primarily based on the paper by Bharath Hariharan and Ross Girshick, 2016 [1]. At a very high level, the method is as follows: given **only few** samples of a novel class, we train a generator network to generate many new 'hallucinated' examples for this same class. We then use those new generated examples, as well

*Tel Aviv University, Tel Aviv, Israel. Correspondence to: Uri Avron <uriavron@gmail.com>, Jontahan Somer <somer@mail.tau.ac.il>, Matan Harel <matan.harel.mh@gmail.com>

as those that were given to us, in order to train another neural network to perform the classification task.

The method for generating new training examples is based on the insight that variation within one category might be transferable to another category. For instance, a certain variation in anatomy may impact the chest images similarly, regardless of the disease.

We elaborate on the points in which we implemented novel solutions and advancements as we dive into the details of the method .

2. Our Approach

The method is composed of 4 parts:

1. Feature representation learning
2. Training a generator
3. Generating data using the generator
4. Training a classifier using the generated data

We first introduce the datasets we used and then describe the method in detail.

2.1. Datasets used

We tested our results on 3 different datasets. The first 2 are the well researched MNIST and CIFAR10 datasets. The third is a new chest X-ray image dataset.

In May 2017, the "ChestX-ray8" dataset was presented by a team of researchers from the NIH [2]. In the paper they present the methods used to generate the data. A short summary of the data is as follows:

- 108,948 images of 32,717 patients.
- 8 disease labels text-mined from radiological reports.
- Each image is labeled with 'Normal' or labelled with more of the 8 disease labels, or
- Labeling: classes are very imbalanced. For example: 84K images were tagged 'Normal' and around 1K were tagged with 'Cardiomegaly '.
- Image sizes are 1024×1024 pixels. These are relatively large images, recall CIFAR images are 32×32. The entire dataset takes around 40GB of space.

Along with the data, they also provide a benchmark for the task of classifying diseases using a deep convolutional neural network (DCNN) they have trained.

2.2. Feature Representation Learning

For the X-ray dataset we used a ResNet50 DCNN pre-trained on the very large and diverse ImageNet dataset, without the last dense layer, in order to generate the features for the images. This method was used in the chest x-ray paper in order to train a classifier on the data and implementing the same method enabled us to first recreate their results and continue from there onto low shot learning.

For the MNIST and CIFAR10 datasets we trained two different CNNs which achieved ~99% and ~90% accuracy on the datasets respectively. In the low-shot-learning setting we do not have access to data from the novel class during representation learning. Thus, we treated each class in turn as the novel class , and trained a classifier on the remaining classes. We then used this classifier, with the last layer removed, as a feature extractor in order to generate features for the novel class as well.

There is a significant difference between the two methods. In our method, representation learning is performed ad hoc for the specific setting, intuitively - “we learn to represent digits by learning to recognize digits”. Whereas, in the first setting a generic network is used to generate features for images from a very specific domain. **As further research we propose to train a DCNN from scratch on the X-Ray dataset and compare the two methods**. We presume that features learnt from data that is close to the domain at hand will be better than those obtained by generic models.

2.3. Learning to Generate New Examples

We now train a generator G for hallucinating images for novel classes. As we have noted, the method for generating new training examples is based on the insight that variation within one category might be transferable to another category. We want to use the knowledge of the intra-class variation of one class in order to generate a diverse set of examples for the novel class.

We train G to “solve analogies”: G will receive as input the concatenated feature vectors $\langle \phi(b_1), \phi(b_2), \phi(x) \rangle$ where b_1, b_2 are two samples from the same base class and x is a novel image. For this input, G will output a vector who solves the analogy $b_1 : b_2 \Rightarrow x : ?$ Thus applying to x the $b_1 \rightarrow b_2$ transformation. Note that the $b_1 \rightarrow b_2$ transformation stays within class B and the generator should perform on x a transformation that does not result in an element of a different class than x (see part 3.3 for further evaluation of the generator’s performance).

G will be a 3 layer multi-layer-perceptron. The training data for G is generated by creating completed analogies - quadruplets of feature vectors from the base classes. We start by clustering each of the base classes into k clusters. Then for each two classes A, B, for each pair of centroids, c_1^A, c_2^A from class A we find the pair c_1^B, c_2^B such that the cosine distance between $c_1^A - c_2^A$ and $c_1^B - c_2^B$ is minimized. Concatenating the 4 centroids results in one element in the dataset.

For each quadruplet $\langle c_1^A, c_2^A, c_1^B, c_2^B \rangle$, we feed the triplet: $\langle c_1^A, c_2^A, c_1^B \rangle$ into G. We want $G(\langle c_1^A, c_2^A, c_1^B \rangle)$ to be as close as possible to c_2^B and also remain within the class B. In order to do so we minimize the loss function:

$$\lambda MSE(G(\langle c_1^A, c_2^A, c_1^B \rangle), c_2^B)$$
$$+(1-\lambda) L_{cls_{BASE}}(G(\langle c_1^A, c_2^A, c_1^B \rangle), B)$$

Where we have MSE the mean square error between the generator's output and the true target and $L_{cls_{BASE}}$ the log-loss of the classifier w.r.t the true class of c_2^B.

2.4. Generating New Examples for a Novel Class

Assume we have received n examples of some novel category. Generating a new example is done by sampling one of these examples - $\phi(x)$, choosing a a base class A , and from it two centroids - c_1^A, c_2^A. We then apply the generator to this triplet and use $G(c_1^A, c_2^A, \langle \phi(x))$ as a new example.

In the paper, the choice of category and centroids was performed uniformly at random. Recall that our goal is to compensate for the lack of intra-class variation contained within the samples of the novel class. Thus, we would like to generate this variation utilizing other classes. We propose a novel way of performing this selection. We attempt to increase the likelihood that the variation within the chosen base class A will be transferable to the novel class. Recall that during the feature representation phase we have trained a classifier over the base classes. We suggest that by applying this classifier on the novel data we will gain some insight into which classes might possibly carry greater transferability to the novel class. To illustrate this: when treating the "cat" class in the CIFAR data set as a novel class, the classifier used in the feature extraction phase was trained over the remaining classes: [airplane, automobile, bird, deer, dog, frog, horse, ship, truck]. Applying this classifier to predict the class of a "cat" example results in "dog" in many cases. It is very likely that the variation in the dog class is much highly relevant to the cat class than for instance the variation in the airplane class. Dogs and cats appear in similar conditions whereas (we hope..) that airplanes and cats do not.

2.5. Training a Classifier on Generated Data

The main consideration here is that the low-shot setting creates a significant class imbalance. In order to fix this we follow the paper and sample the training data uniformly over classes and uniformly within classes.

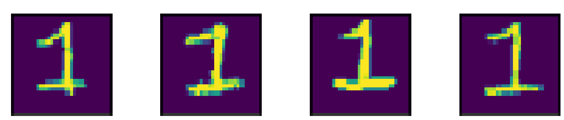
3. Experiments

3.1. Evaluating Clustering over Feature Space

One of the assumptions behind this model is that clustering over feature space results in capturing the variation in the data. We used the K-Means algorithm in order to cluster the data. Recall that K-Means produces centroids that are not necessarily in the original set of points. So in order to test if these centroids capture a variation in the data we plotted for first found for each centroid the 4 examples (features) closes to it. We then plotted the 4 images who resulted in these 4 examples. Figures 1 and 2 validate the assumption that the clustering grasps the intra class variation. Note the total number of clusters was 30 and we randomly chose 4 to display. The results in figures 1,2 also served as a validation that our feature extractor captured features that are also meaningful in "human" terms.

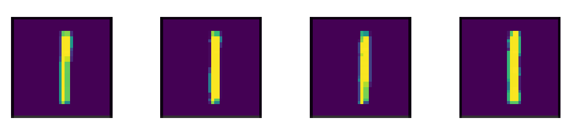
Figure 1. Clustering over Feature Space Captures Intra-class Variation - Digits

Cluster1:



Cluster2:

Cluster3:



Cluster4:

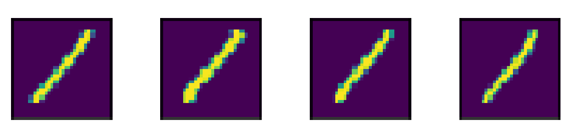
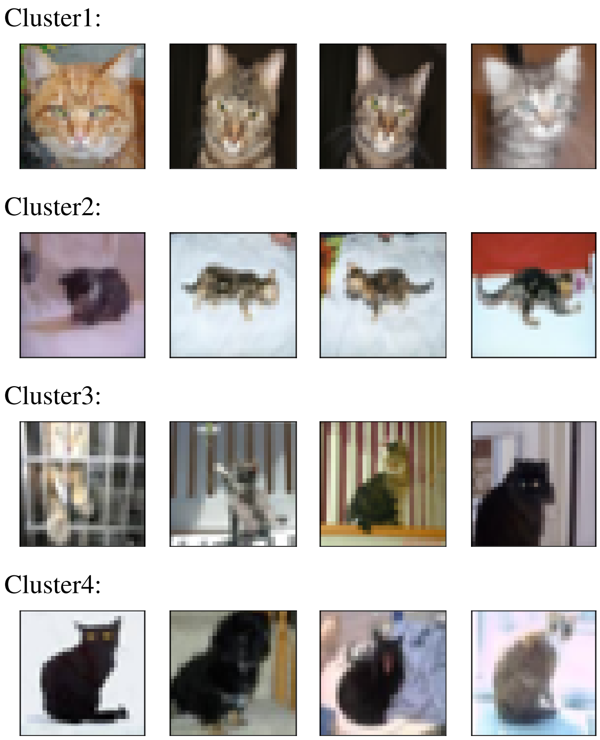


Figure 2. Clustering over Feature Space Captures Intra-class Variation - Cats



3.2. Connection Between Number of Clusters Used in Training and Test Accuracy Achieved

Our generator is intended to perform transformations which move "one centroid to another", thus creating samples with greater variability for the novel class. There is an underlying assumption here that we must have data from different clusters in order to capture the variation and perform well on the test data. We wanted to test this assumption before testing the use of generated data. We first plot for each dataset the accuracy achieved by number of unique samples used for training. Figures 3 and 4 show that the growth is logarithmic in the number of samples. We now fix the sample size and test how the number of clusters from which the training data is taken affects the accuracy. We chose to use 256 samples as we have seen that the returns in accuracy are diminishing from that point forward. Figures 5 and 6 show that taking data from more centroids does indeed increase the models capacity to generalize on the test set. All results are averaged over 5 runs.

Figure 3. Accuracy Achieved by Number of Unique Samples Trained On - CIFAR

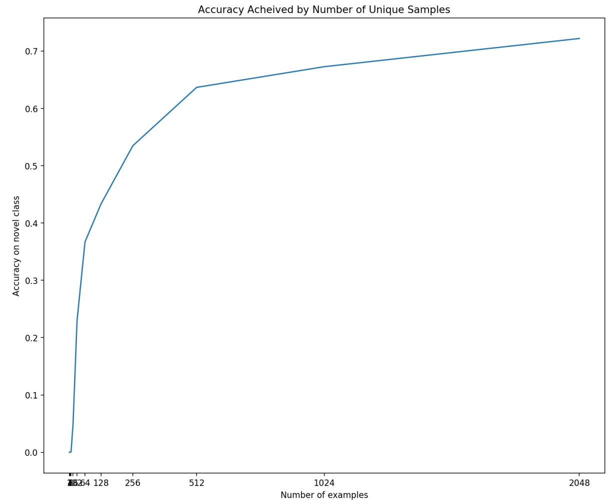
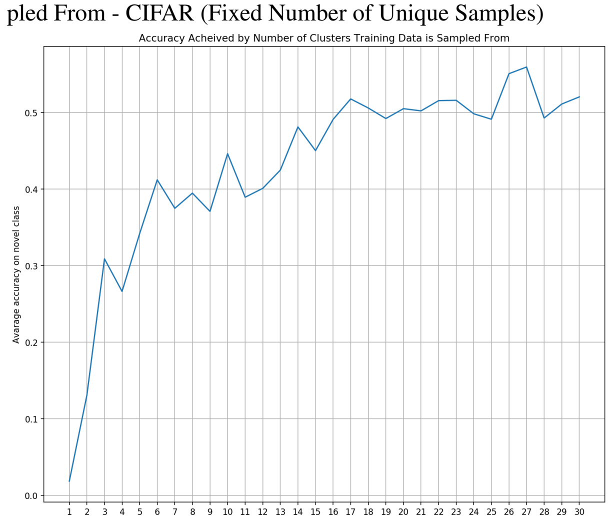


Figure 4. Accuracy Achieved by Number of Clusters Data is Sampled From - CIFAR (Fixed Number of Unique Samples)



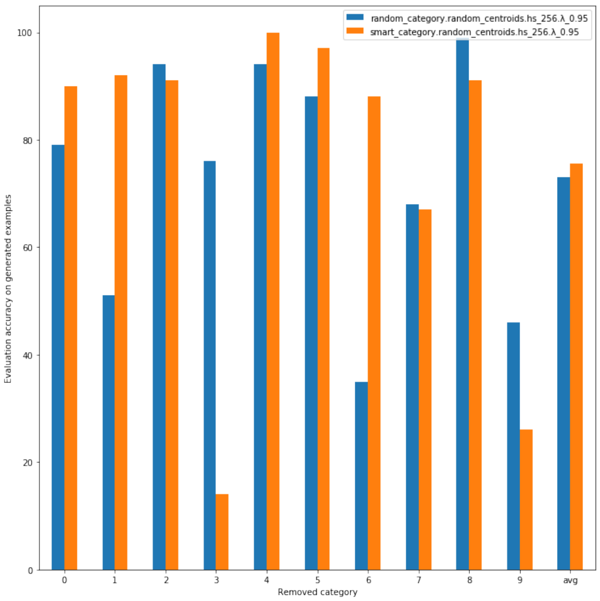
3.3. Evaluating the Performance of the Generator

In essence, the generator must perform well on two distinct tasks.

3.3.1. Task 1: Transformations Generate New Instances of the Same Class

Given a sample from a novel class the generator must generate samples which are from the same class. This is non-trivial since it has never trained on data from this class. There is an underlying assumption that training the generator to perform transformations which leave the base classes in the same class will result in the generator performing similarly on the novel class. As the generated

Figure 5. Accuracy Acheived on Generated Data + Comparison of Methods - MNIST



examples do not correspond with real images, we devised the following test to check if they are from the correct class: we first train a classifier on all examples of the base and novel classes, this classifier serves as our oracle. We expect that this classifier will recognize the generated images from samples of the novel class as instances of the novel class.

At this point we also provide a comparison between the standard method of generating data - choosing a category at random, and our "smart selection" of a category. We show that this new method provides some improvement over the standard method of generation. Figure 5 plots this comparison - random category selection in blue and "smart selection" in orange. The X axis corresponds with the class removed while training the generator - this is the novel class. Note the rightmost column which depicts the average accuracy over all categories. We also provide the parameters we used: The number of neurons used for the generators hidden layers: 256, and the parameter $\lambda = 0.95$ used in the loss function. These were discovered via cross validation. We fix these parameters from here on

3.3.2. Task 2: Samples Generated have Significant Variation

Notice that this first task could be achieved trivially by returning the sample received as it is. If the oracle classifier recognizes this sample to be from its correct class, this trivial method will achieve 100% accuracy in the previous test we provided. Of course this will not provide for any additional capacity for generalization. Thus, we must also evaluate the generators ability to induce variation in the generated samples. We do this by clustering all of the novel class's data and assigning each generated example to a cluster. As we have seen before, using data from more clusters provides for better generalization capacity.

1. describe the pipeline experiment. pit falls etc. the most we can squeeze out of the generator
2. figures and results.

References

[1] Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. *arXiv*, 2016.

[2] Xiaosong Wang. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks. *National Institutes of Health*, 2017.