# Progress Report
# 1-15 November 2012

Amin Karbasi
E-mail: amin,karbasi@epfl.ch
Amir Hesam Salavati
E-mail: hesam.salavati@epfl.ch

Supervisor: Prof. Amin Shokrollahi
E-mail: amin.shokrollahi@epfl.ch

Algorithmics Laboratory (ALGO)
Ecole Polytechnique Federale de Lausanne (EPFL)

# 1 Summary

In the past two weeks our main focus was to address the comments of the reviewers on our ICML paper. To this end, we had to implement and test various ideas for extracting features from a dataset of natural images. Furthermore, we unfortunately found out our simulations on the dataset of spoken english words were also incorrect due to an error in pre-processing the dataset. Thus, we spent most of our time on feature extraction techniques. We have tested a few different approaches the details of which is given in the rest of the report.

# 2 Introduction

Recently, we have been trying to apply our algorithm to learn images in the STL-10 (and CIFAR-10) dataset(s). We have found out that the algorithm could learned some dual bases over *quantized version* of the pixel-level images or the features extracted from the images. However, **it can not classify**, i.e. either the learned dual vectors are not orthogonal to the test dataset of the same class or they are orthogonal to the dataset of other classes. In both cases, we can not classify using our approach. Thus, we will focus only on the **associative memory** application, i.e. recalling already memorized patterns.

However, for our approach to be meaningful, we must memorize a large number of patterns. So, since our images have the size of $96 \times 96$, we must either find a dataset that has much more than 9216 images or use feature extraction and pooling techniques to reduce the number of parameters to a couple of hundreds. We will focus on the second approach throughout this report.

# 3 Learning dual bases for clusters with various cluster and pooling sizes

To reduce the number of variables to learn, we reduce the number of parameters by pooling over the extracted features. Here, the pooling window affect the selected features while the cluster size affect the number of nodes in each cluster. Roughly speaking, if we have more neurons, there is a greater chance of learning some dual bases. However, as the cluster size increases, the performance of the recall algorithm decreases. Therefore, we have a trade off and should find a balance between the learning and recall algorithm performances.

## 3.1 Pooling window: $5 \times 5$; Cluster size $6 \times 6$

We have tested this method with two different options: learning over quantized features and learning over quantization indices.

**Result for quantization indices:** Failure. We couldn't learn dual bases or even if we could learn a few, they are all the same, i.e. we learn redundant constraints. So we either have to give up working with quantization indices or find a set of quantization steps that result in the success of the learning algorithm.

**Result for quantization values:** So so. This time, we could learn a few bases but they are not enough for the recall algorithm to work. Furthermore, may of the pattern neurons in the cluster were unconnected in the end to any constraint node. In addition, a later inspection lead us to the conclusion that the convergence of the learning algorithm in this case is manily due to the fact that the feature values are small. Thus, we could easily find some vectors that result in *low* projection values for the patterns in the dataset. A negative aftereffect of this phenomenon would be the fact that patterns from any class will be orthogonal to the learned matrix of the other classes as well!

#### 3.1.1 Discussion

The problem with the approach in this section is that the rank of the part of the dataset that lies withi the domain of our cluster was 35 out of a maximum of 36. As a result, it might be difficult to find even approximate dual bases (although in some other cases we could find *approximate* bases in similar circumstances).

### 3.2 Pooling window: $5 \times 5$; Cluster size $10 \times 10$

**Result:** Failure. Even though we increased the number of nodes in a cluster, the results were not noticeably better from the previous case.

### 3.3 Random clustering with quantization indices for the CIFAR-10 dataset[1]

**Result:** Failure. The algorithm only finds one or two constraints in a cluster with 76 nodes.

## 4 Effects of quantization methods on the learning algorithm

To overcome the issues mentioned in the previous section, we have also tested a few different tweaks on the quantization techniques we used. The results are given in the sequel.

### 4.1 Working with only positive quantization levels

**Result:** Failure. Even though the algorithm could learn a few constraints, but overall it can not learn a sufficient amount of constraint for the recall algorithm to work properly.

### 4.2 Working with coarser quantization levels

**Result:** So so. This approach was capable of learning a rather good amount of bases. But this is only achieved since the rank of the dataset of sub-patterns is very low. Nevertheless, since there are no equal columns in the learned matrix, the algorithm will be capable of correcting a single error.

However, the main problem with this approach is the fact that many of the sub-patterns will be the same if the quantization is coarse. Albeit this might not be an issue overall since what is important is that the patterns are distinguishable overall and not only their sub-patterns. So even if in one clusters two patterns look alike, that will not necessarily put us in trouble. In fact, this is the case with the following quantization levels:

[1] Random clustering technique is explained in a later section.

- $[0.025 : 0.025 : .2]$
- $[0.05 : 0.05 : .2]$

#### 4.2.1 Discussion

Another important issue with using quantization levels is that they are not necessarily integers. Therefore, our recall algorithm will not work with them, at least in its current format.

# 5 The spoken English word dataset

As mentioned earlier, we realized that there was an error in our previous simulations over the dataset of spoken English words and as a result, we must repeat out steps. To this end, we processed the dataset again and considered three different quantization methods over the values in the frequency domain: *fine*, *coarse* and 0/1. Here are the results:

## 5.1 Fine quantization

**Result:** Failure. We could barely learn any constraints.

## 5.2 Coarse quantization

**Result:** Failure. It was better than the previous case but still was not acceptable.

## 5.3 0/1 quantization or thresholding

**Result:** So so. The learning algorithm worked in this case where we apply a threshold to the valyes and if they are larger than that threshold we count them as 1, and otherwise, as 0.

However, the main reason that this technique works is that due to this thresholding, most of the entries, specially in the high frequencies, are zero. And finding a vector that is orthogonal to a set of all-zero sub-patterns is not really that difficult! Even worse, we could not find a constraint that involves many of the non-zero entries of the sub-patterns. Consequently, we ended up with a lot of pattern nodes that are not connected to any constraint at all, which of course means that we can not correct any errors in that part of the sub-patterns in the recall algorithm.

## 5.4 Random clustering with quantization indices[2]

**Result:** Failure. The algorithm either does not converge or is very slow.

## 5.5 Random clustering with quantized values[3]

**Result:** Failure. The values are small so they will look orthogonal to any random vector in our learning algorithm (in its current format).

[2]Random clustering technique is explained in a later section.
[3]Random clustering technique is explained in a later section.

## 6 Random clustering techniques for learning features from natural datasets

Another technique that we experimented with was to determine clusters in a random fashion. So far, we had naturally select clusters such that they contain neighboring features in the feature map of an image. However, in the random clustering technique, we put all the features in a long vector together and randomly selected different features to include in each cluster.

Furthermore, we did not fix the size of the cluster either and for each cluster, first determined its size randomly (according to a Gaussian $\mathcal{N}(\mu, \sigma^2)$ distribution) and then picked the same number of pattern neurons to include in the cluster. When the average cluster size $\mu$ is small, we do not enforce the sparsity constraint within the cluster and just have to make sure that we have global sparsity.

We then applied the learning algorithm to each cluster. Once finished, had we learned at least a few constraints, we continue to the next step and, otherwise, ignore the results. Next, we put all the learned constraints in a big matrix, corresponding to the connectivity graph of the whole network. At this point, we divide the network into a different set of random clusters which will be used for the recall phase. This last step could be viewed as a scheduling method that decides which part of the network should update at any moment of time in the recall phase.

## 7 Conclusions and future work

We have tested a few approaches in extracting features. Unfortunately, most of them did not result in a suitable representation for our learning algorithm. The result of the learning and feature extraction techniques that we have tested are available online in [1] and [2], respectively. We will nevertheless continue our efforts to find an appropriate feature extraction technique.

## References


[1] A. H. Salavati, A. Karbasi, *The convergence status of dual space algorithm over various databases*, Available online at `https://docs.google.com/spreadsheet/ccc?key=0AmpoqrkA0LCydHJSV1FPMzZ6U1U2eEh2cFdqS2huWkE#gid=0`

[2] A. H. Salavati, A. Karbasi, *Feature extraction steps for various databases*, Available online at `https://docs.google.com/spreadsheet/ccc?key=0AmpoqrkA0LCydGVoMzJpU3F0c0FrX1RnZGFfWW40Nmc#gid=0`