Paper SAS3100-2016

An Efficient Pattern Recognition Approach with Applications

Patrick Hall, Jorge Silva, Ilknur Kaynar Kabul, Keith Holdaway, and Alex Chien
SAS Institute Inc.

ABSTRACT

This paper presents supervised and unsupervised pattern recognition techniques that use Base SAS® and SAS® Enterprise Miner™ software. A simple preprocessing technique creates many small image patches from larger images. These patches encourage the learned patterns to have local scale, which follows well-known statistical properties of natural images. In addition, these patches reduce the number of features that are required to represent an image and can decrease the training time that algorithms need in order to learn from the images. If a training label is available, a classifier is trained to identify patches of interest. In the unsupervised case, a stacked autoencoder network is used to generate a dictionary of representative patches, which can be used to locate areas of interest in new images. This technique can be applied to pattern recognition problems in general, and this paper presents examples from the oil and gas industry and from a solar power forecasting application.

INTRODUCTION

Pattern recognition in images has received renewed attention in recent years because of increasingly abundant video data and new advances in image processing and machine learning. Specifically, new methods for training deep neural networks that have a large number of parameters are now available and have been successfully applied to data sets of millions of images (Krizhevsky, Sutskever, and Hinton 2012). In addition, theoretical results in sparse signal processing now yield performance guarantees regarding dimensionality reduction in natural images (Kreutz-Delgado et al. 2003).

These advances in deep learning and sparse signal processing have made it possible to solve previously intractable business problems, such as detecting objects of interest in long video sequences and finding a small and interpretable set of patterns to accurately represent large volumes of data. The aim of this paper is to describe some of these new methods and show you how to implement them.

First, this paper describes a simple technique for decomposing an image into *patches*, which are meaningful blocks of sufficiently small size for efficient processing. For natural images, patches enable significant dimensionality reduction while capturing relevant local information from different regions of the image of interest.

Second, the paper introduces the concept of a *dictionary*, which is a generalization of a basis in a vector space. For the purpose of this paper, a dictionary is a relatively small set of representative patches, which are important because they enable feature extraction in an interpretable way and can be learned in an unsupervised manner (that is, in the absence of training labels). Dictionary learning for images is an important problem in its own right and constitutes the foundation of several state-of-the-art compression and reconstruction methods (Lee et al. 2006).

Third, the paper reviews the main concepts behind deep learning, with a particular focus on unsupervised pretraining that uses a type of neural networks called *stacked autoencoders*. As demonstrated in prominent examples in the literature (Bengio, Courville, and Vincent 2013), stacked autoencoders are well suited for unsupervised learning of dictionaries. Although the dictionaries themselves are of great interest, the paper briefly introduces *k*-means clustering and deep neural network (DNN) classification methodologies for continued analyses of the raw image patches and the image features that are extracted by the denoising autoencoder.

Fourth, the paper uses real business cases to show how to implement these techniques and discusses the results. The unsupervised case, in which no labels are provided for training, is illustrated by learning a dictionary of interpretable patterns and finding clusters of image patches that can help oil and natural gas producers automatically locate interesting features in seismic images. The supervised case, in which labels can be used to train DNNs for classification or regression applications, is demonstrated by classifying image patches from total sky image videos as cloudy or sunny weather in order to help renewable energy operators predict power output. Example Python code is provided to create image patches and decode image patches into comma-separated variable (CSV) format. SAS code is supplied for reading the image patches from CSV format, creating dictionaries and clusters of image patches, and using DNNs for supervised learning tasks. Code and other supplemental materials are available through the SAS organizational GitHub repositories:

https://github.com/sassoftware/enlighten-apply/tree/master/SAS_Py_Patches_PatternRecognition

Finally, the paper proposes directions for future work, including evaluation of convolutional neural networks (CNNs) for classifying images, broader application of deep learning for hydrocarbon reservoir discovery, direct forecasting of sun irradiance using image features, and evaluation of a new feature engineering technique that is invariant to translation, rotation, and scale.

METHODS AND TECHNOLOGIES

DECOMPOSING AN IMAGE INTO PATCHES

Consider an image I of size $H \times W$ pixels. The pixel at the ith row and jth column of the image is denoted $I(i,j)$. In the case of a gray-scale image, $I(i,j)$ has an intensity value, which is usually an integer between 0 and 255. In the case of a color image in the red-green-blue (RGB) coordinate system, $I(i,j)$ is a tuple that consists of three integer values: $I_R(i,j)$, $I_G(i,j)$, $I_B(i,j)$. These values are also usually between 0 and 255.

A gray-scale image of resolution 2560 x 1920 consists of nearly 5 million pixels, a prohibitive number of features for the vast majority of predictive algorithms to use directly as input. A common practice in image processing consists of dividing the image into square subimages, also called *patches* or *tiles*. Let P be a patch of size $B \times B$. For example, if $B = 20$, then P contains 400 pixels, which is far more suitable for analysis. In general B is an important tuning parameter, and you should choose B large enough to capture recognizable patterns in the image. Patches are often allowed to overlap in order to avoid *block artifacts* (visual discontinuities that can occur along the boundaries of patches after they are processed). In Figure 1, patches P and P' have size $B \times B$, and they overlap.

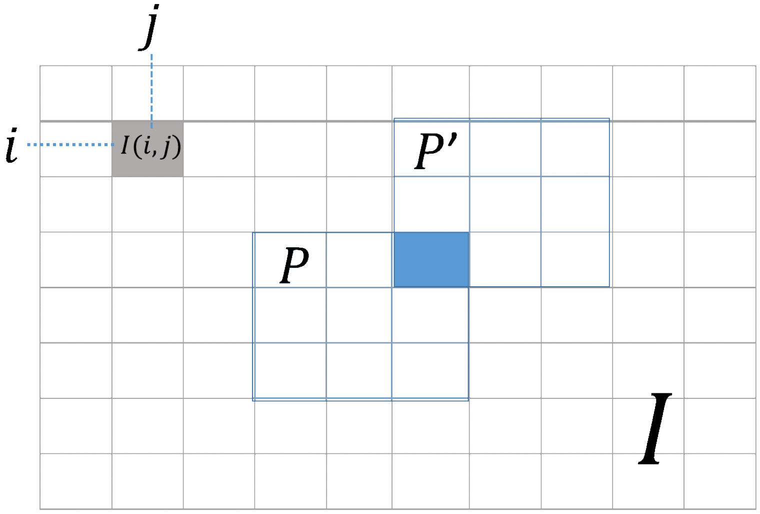

Figure 1. Illustration of overlapping patches in an image.

For natural images, patches can be efficiently represented as linear combinations of elements, which are called dictionary atoms. The number of dictionary atoms depends on B. You can vectorize a patch by rearranging all the pixels into a row vector of size B^2, denoted $\boldsymbol{x} = vec(B)$. See Figure 2 for an example.

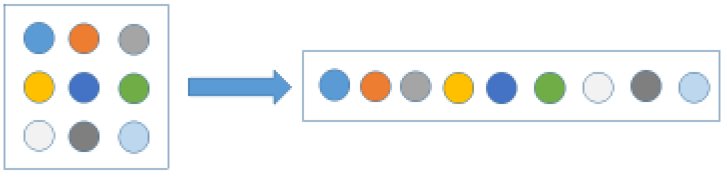

Figure 2. Example of patch vectorization. Pixels are rearranged in a row vector in lexicographic order.

REPRESENTING PATCHES WITH A DICTIONARY

Like any vector of dimension B^2, $vec(B)$ can be represented as a linear combination of B^2 orthonormal basis elements. Instead of a basis, this paper uses a *dictionary*, which is a generalization in which the elements (called *atoms*) are not required to be orthonormal. The advantage of a dictionary over a basis is that dictionaries often yield a more efficient representation. Figure 3 shows an example of a dictionary. Each atom is itself a subimage that has the same dimensions as a patch has.

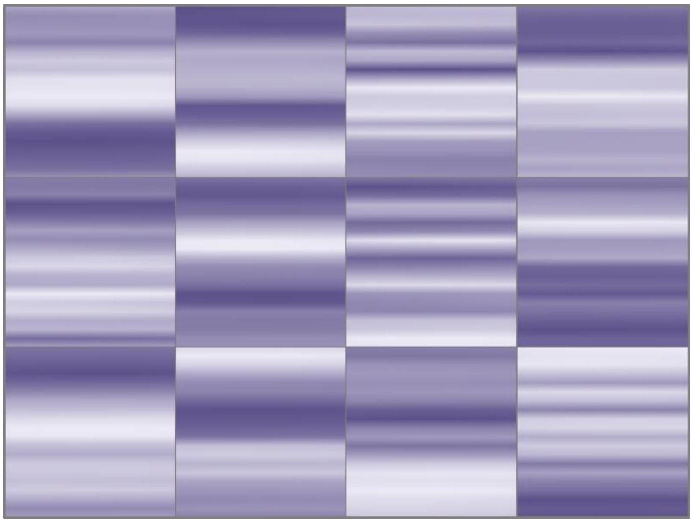

Figure 3. This example of a dictionary has 12 atoms (arranged in a grid for visualization only).

The mathematical expression for the representation of a patch P is

$$\boldsymbol{x} = vec(P) = \sum_j \alpha_j \boldsymbol{d}_j$$

where each α_j is a scalar coefficient and each $\boldsymbol{d}_j$ is a vectorized dictionary atom.

A good dictionary should contain atoms that are themselves similar to the image patches, so that the atoms can be represented using as few nonzero coefficients as possible. Such representations are considered *sparse* and constitute a powerful tool for dimensionality reduction and image summarization. Dictionaries can be learned via stacked autoencoder neural networks, as described in the following section.

STACKED AUTOENCODERS

Neural networks are a powerful modeling tool. This paper focuses specifically on stacked autoencoders, which perform unsupervised feature extraction. Briefly, the network is trained by setting the target neurons equal to the input neurons. There are multiple layers and a bottleneck in the middle layer, so the network is forced to learn a reduced-dimensional internal representation of the inputs before reconstructing them in the output layer. The weights in the output layer have the same dimensionality as the patterns themselves, and they constitute the dictionary, as shown in Figure 4. You can find more thorough descriptions of deep neural networks and stacked autoencoders in Bengio, Courville, and Vincent (2013) and of how the NEURAL procedure in SAS Enterprise Miner can be used to train autoencoders in Hall et al. (2014).

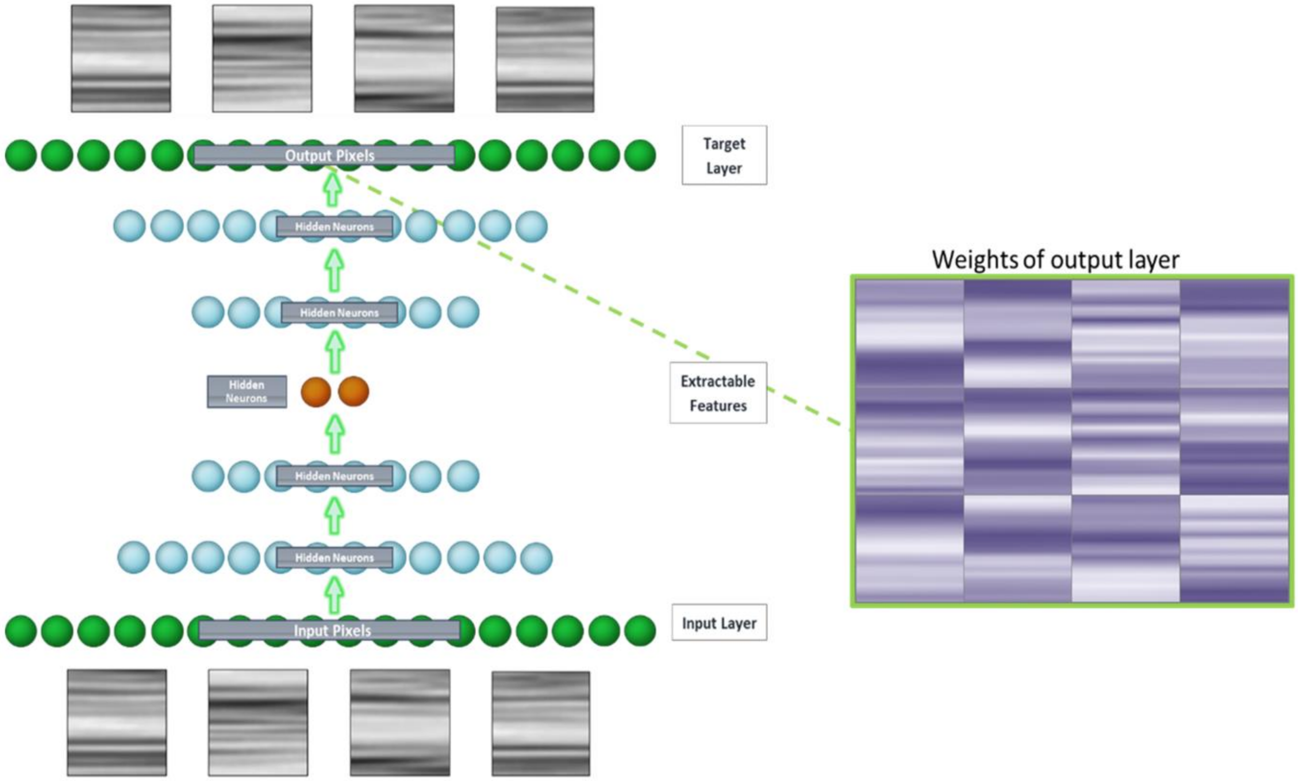

Figure 4. Schematic of a stacked autoencoder. Outputs are set equal to input patterns, and the learned weights of the output layer form the dictionary.

To gain additional information about the original input images in an unsupervised manner, you can perform cluster analysis on the image patches by using the *k*-means algorithm, which is available in the HPCLUS procedure in SAS Enterprise Miner. For improved computational efficiency, cluster analysis can also be performed in the low-dimensional projections that are created by the middle hidden layer of an autoencoder. In either case, the best number of clusters, *k,* can be estimated by the aligned box criterion (ABC). In addition to using unsupervised training, you can also solve supervised classification or regression problems by using deep neural networks whenever targets are available in the training data set.

RESULTS

Two experiments were performed to substantiate the unsupervised and supervised aspects of this patch-based approach to image analysis. For the unsupervised tasks, a dictionary of representative image patches was created by a deep neural network from high-resolution seismic images used for hydrocarbon reservoir discovery. In a separate unsupervised task, the same image patches were clustered by the *k*-means algorithm using the aligned box criterion to automatically estimate the best number of clusters. Example high-resolution seismic images are presented in Figure 5.

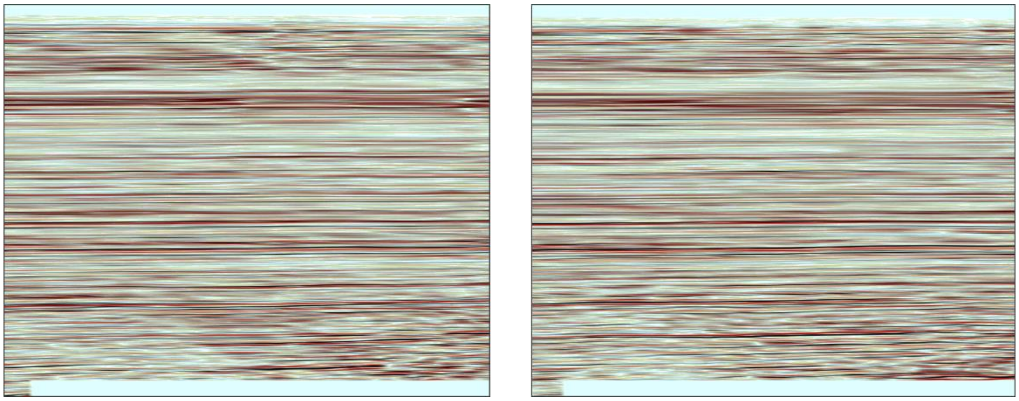

Figure 5. High-resolution seismic images.

UNSUPERVISED RESULTS

Twenty-four high-resolution seismic images were converted to gray scale and split into 50 x 50 pixel patches by using a 25-pixel stride length. Patches were then downsampled to 20 x 20 pixels to increase processing efficiency. The patching process resulted in a training set that contains 30,459 input subimages like those in Figure 6.

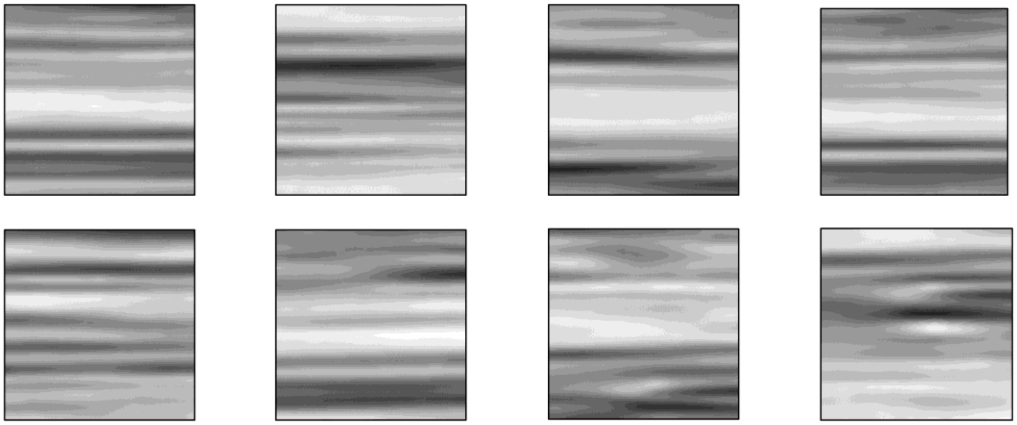

Figure 6. Sample 20 x 20 pixel patches created from high-resolution seismic images.

The NEURAL procedure was used to train a five-layer autoencoder with 50 hidden units in the first layer, 25 hidden units in the second layer, 10 hidden units in the third layer, 25 hidden units in the fourth layer, and 50 hidden units in the fifth layer (50-25-10-25-50). Because the autoencoder contained 50 hidden units in the fifth and top output layer, 50 dictionary images were created. Examples from this dictionary are illustrated in Figure 3.

Following Hall et al. (2014), the autoencoder was initialized using layerwise pretraining with the FREEZE and THAW statements in the NEURAL procedure. All layers were then simultaneously trained to convergence. For efficiency, conjugate gradient optimization is used when training all layers of the network together. The TECH=CONGRA option in the TRAIN statement is used to specify the optimization method:

```
/* final training of all layers using PROC NEURAL */
thaw i->h1;
thaw h1->h2;
thaw h2->h3;
thaw h3->h4;
train tech=congra;
```

Input patches were also clustered to reveal different segments of interest in the original input images. The HPCLUS procedure was used to cluster the input patches by the *k*-means algorithm. The ABC method, with a criterion of the first peak value among all the peak values of the gap statistic, was used to estimate four as the best number of clusters for the input data. In the following statements, the ABC method is specified in the NOC option in the HPCLUS statement, and the ABC selection criterion is specified in the CRITERION= option:

```
/* automatically estimate the best number of clusters using ABC */
proc hpclus
  data=train
  maxclusters=20
  noc=abc(b=3 minclusters=2 align=PCA criterion=firstpeak)
  maxiter=1000;
  input pixel_: / level=interval;
run;
```

The *k*-means algorithm usually results in spherical clusters. However, clustering patches (as opposed to entire images) can result in complex clusters in the original image as can be seen in Figure 7.

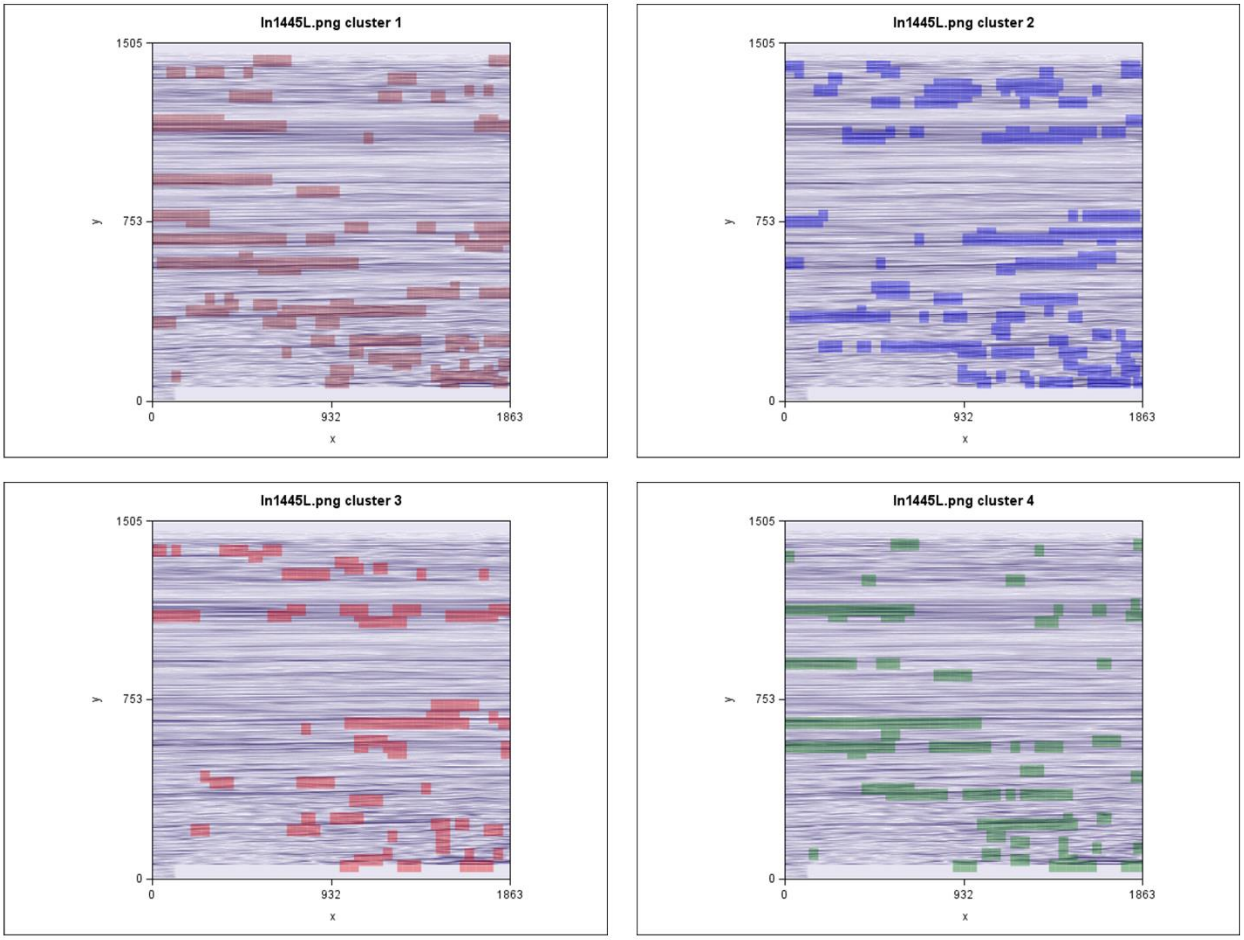

Figure 7. Four clusters of patches overlaid onto a single inline, two-dimensional seismic image.

SUPERVISED RESULTS

For the supervised task, 360-degree total sky images that are used for a renewable energy application were classified into two categories (sunny or cloudy) by using a deep neural network. The images are available by request from the Atmospheric Radiation Measurement Program archive (ARM 2015). Example sunny and cloudy 360-degree total sky images are presented in Figure 8.

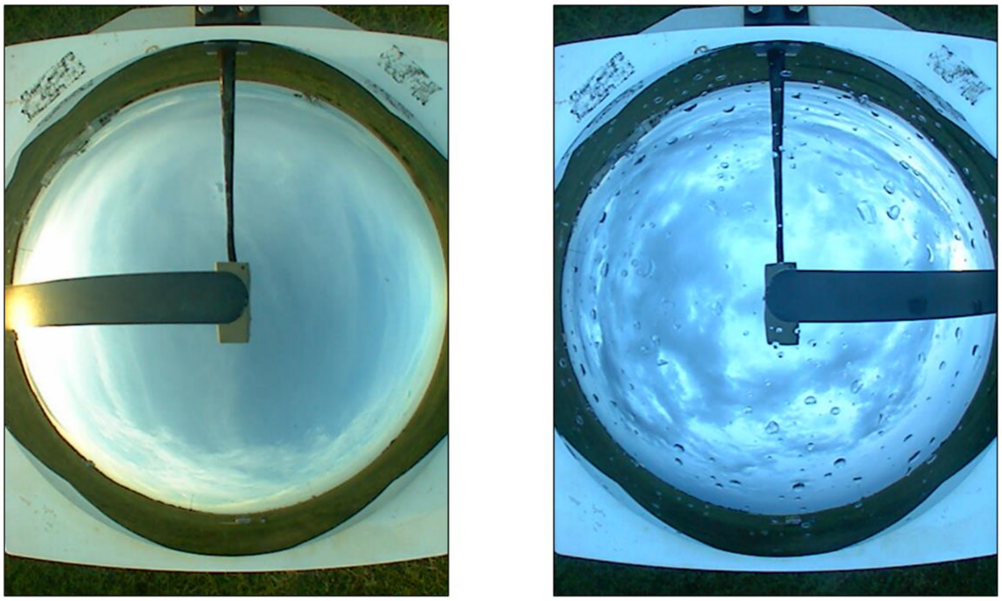

Figure 8. Two 360-degree panoramic images of a sunny sky (left) and a cloudy sky (right).

A sample of 360-degree total sky images were converted to gray scale, divided into 30 x 30 pixel patches, and further downsampled to 25 x 25 pixel patches. Figure 9 displays a sample of 8 of the 42,998 resulting subimages in the training set.

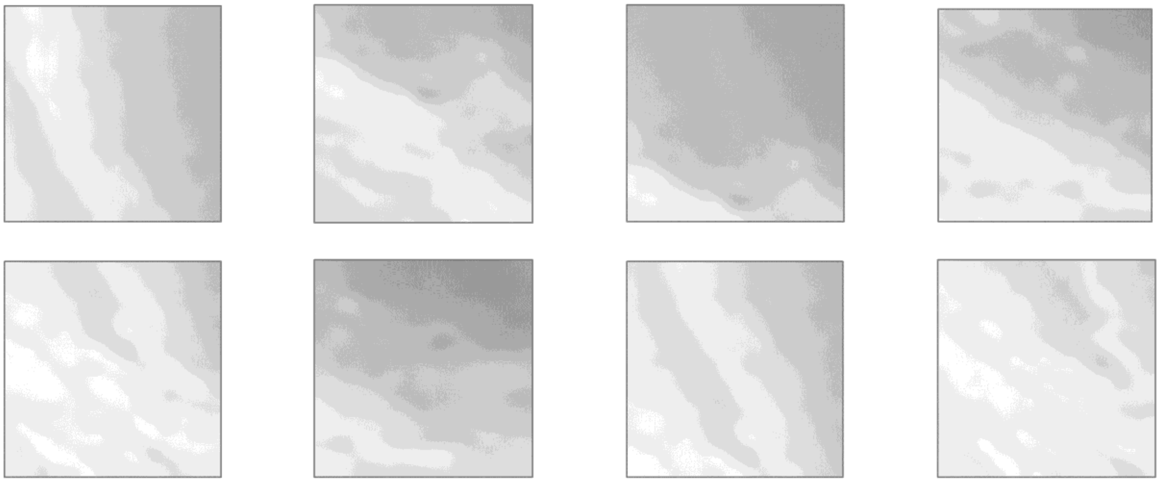

Figure 9. Sample 25 x 25 pixel patches created from 360-degree panoramic images.

The NEURAL procedure was used to train a three-layer (200-100-50) deep neural network (DNN) on the image patches, and the DNN was initialized by layerwise pretraining. All layers were then subsequently trained together to convergence by using conjugate gradient optimization. Based on labels provided by the ARM archive, patches were classified as either sunny or cloudy. Patches were classified accurately by the DNN and can be used to assign original images as proportionally sunny or cloudy as presented in Figure 10. Such classification has immediate application in forecasting solar- or wind-power generation.

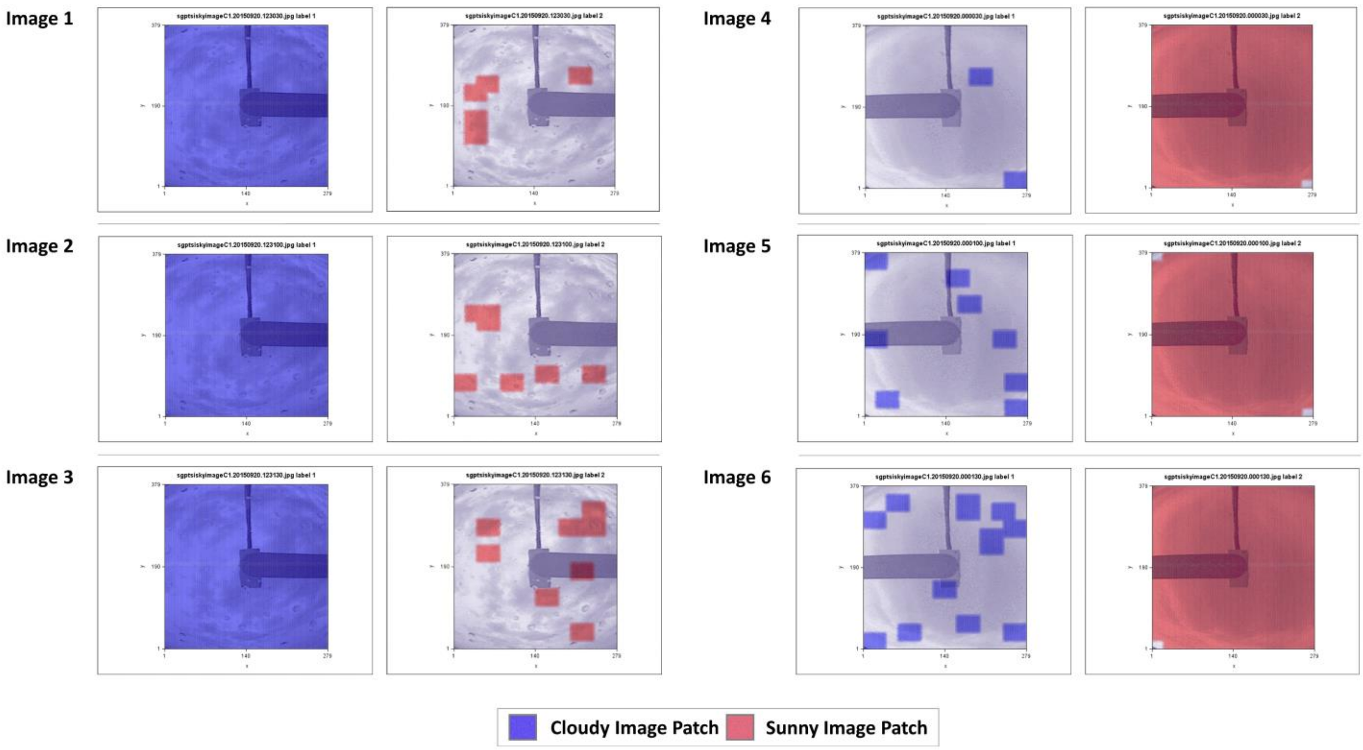
Image 1
sgptsiskyimageC1.20150920.123030.jpg label 1
sgptsiskyimageC1.20150920.123030.jpg label 2
Image 2
sgptsiskyimageC1.20150920.123100.jpg label 1
sgptsiskyimageC1.20150920.123100.jpg label 2
Image 3
sgptsiskyimageC1.20150920.123130.jpg label 1
sgptsiskyimageC1.20150920.123130.jpg label 2
Image 4
sgptsiskyimageC1.20150920.000030.jpg label 1
sgptsiskyimageC1.20150920.000030.jpg label 2
Image 5
sgptsiskyimageC1.20150920.000100.jpg label 1
sgptsiskyimageC1.20150920.000100.jpg label 2
Image 6
sgptsiskyimageC1.20150920.000130.jpg label 1
sgptsiskyimageC1.20150920.000130.jpg label 2
379
190
1
140
279
y
x
Cloudy Image Patch
Sunny Image Patch

Figure 10a. Sample cloudy images with most patches classified as cloudy.

Figure 10b. Sunny images with most patches classified as sunny.

CONCLUSION

This paper proposes a framework that enables you to apply pattern recognition methods in images by using Python, Base SAS, and SAS Enterprise Miner. By decomposing an image into patches, you can overcome the problem of high dimensionality in the image and the problem of making the learned patterns invariant to translations. By following this methodology, you can use the NEURAL procedure in SAS Enterprise Miner to learn a dictionary for efficiently representing the patches in a reduced-dimensional space. Moreover, you can find groups of related patches through scalable *k*-means clustering, which is available in the HPCLUS procedure in SAS Enterprise Miner. In addition to this unsupervised analysis, you can also use PROC NEURAL in a supervised learning setting when a target variable is available.

This approach is demonstrated on real business problems from traditional and renewable energy industries, and highly encouraging results are obtained. Future work will proceed in several directions. Comparisons with convolutional neural networks and other state-of-the-art methods will be undertaken. Deep learning techniques might be applied more broadly to oil and gas reservoir characterization where extremely large seismic datasets can be analyzed to identify subtle stratigraphic traps in addition to repeatable wavelet patterns representative of hydrocarbon flows through porous rock. For solar power output forecasting, a direct measure of the observed sun irradiance or a ratio of the observed sun irradiance to the theoretical sun irradiance under a totally clear sky will be modeled. A trained model using irradiance measures could then be applied to different locations and seasons. Finally, a novel image preprocessing technique that promotes the creation of scale-, translation-, and rotation-invariant features is currently being researched.

REFERENCES

Atmospheric Radiation Measurement Program. “ARM Archive.” Accessed February 1, 2016. http://ncvweb.archive.arm.gov/.

Bengio, Y., Courville, A., and Vincent, P. (2013). “Representation Learning: A Review and New Perspectives.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35:1798–1828.

Hall, P., Dean, J., Kabul, I. K., and Silva, J. (2014). “An Overview of Machine Learning with SAS Enterprise Miner.” In *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings14/SAS313-2014.pdf.

Kreutz-Delgado, K., Murray, J. F., Rao, B. D., Engan, K., Lee, T.-W., and Sejnowski, T. J. (2003). “Dictionary Learning Algorithms for Sparse Representation.” *Neural Computation* 15:349–396.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). “ImageNet Classification with Deep Convolutional Neural Networks.” In *Advances in Neural Information Processing Systems 25* (NIPS 2012), Proceedings of the Annual Conference on Neural Information Processing Systems, 1097–1105.

Lee, H., Battle, A., Raina, R., and Ng, A. Y. (2006). “Efficient Sparse Coding Algorithms.” In *Advances in Neural Information Processing Systems 19* (NIPS 2006), Proceedings of the Annual Conference on Neural Information Processing Systems, 801–808.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors:

Patrick Hall
SAS
Patrick.Hall@sas.com

Jorge Silva
SAS
Jorge.Silva@sas.com

Ilknur Kaynar Kabul
SAS
Ilknur.KaynarKabul@sas.com

Keith Holdaway
SAS
Keith.Holdaway@sas.com

Alex Chien
SAS
Alex.Chien@sas.com