Biologically inspired feature creation for multi-sensory perception

Brandon ROHRER [a,1]
[a] *Sandia National Laboratories, USA*

Abstract. Automatic feature creation is a powerful tool for identifying and reaching goals in the natural world. This paper describes in detail a biologically-inspired method of feature creation that can be applied to sensory information of any modality. The algorithm is incremental and on-line; it enforces sparseness in the features it creates; and it can form features from other features, making a hierarchical feature set. Here it demonstrates the creation of both visual and audio features.

Keywords. feature creation, unsupervised learning, perception, abstraction, vision processing, sensor fusion

1. Introduction

The purpose of this work is to demonstrate feature creation in multiple sensory modalities that is inspired by observations of neurological activity and structure. A feature is a combination of inputs that is useful in sparsely representing a state variable or in predicting future inputs. Feature creation[2] is most often found to be useful in machine learning problems with high-dimensional inputs and has received increased attention in recent years in problems with large input spaces, such as gene selection and natural language analysis. [5] The problem addressed here is multisensory perception, as in an autonomous robot.

The power of features is well illustrated by a simple example, the exclusive-or (XOR) problem. (See Figure 1.) As posed, the two classes are not linearly separable, that is, they are not separable by a single straight line and cannot be segregated in a two-layer neural network. [9] However, a nonlinear feature, created by multiplying the values of x_1 and x_2 for each data point, transforms the classification problem to a space in which it is trivial.

The ability of appropriately selected features to simplify otherwise intractable problems makes the approach appealing for challenging machine learning applications. Identifying an object category, based on an image, or a topic, based on a text snippet, are popular problems in which constructed features are commonly used. Deciding that an array of pixels represents, say, a dog, may involve several levels of features. Pixels may be in-

[1]Corresponding Author: Sandia National Laboratories, Albuquerque, New Mexico, USA, Email: brrohre@sandia.gov, Web: http://www.sandia.gov/~brrohre

[2]Synonymous and closely related terms include feature extraction, feature construction, feature selection, feature generation, concept generation, sparse representation, kernel learning, manifold detection, or state space dimensionality reduction.

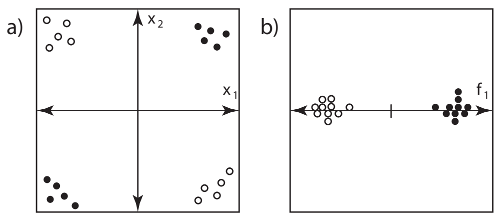

Figure 1. **a)** The exclusive-or problem, named for the XOR logic table it resembles. The challenge is to separate the data points into their respective classes. It clearly cannot be achieved by a linear separation boundary. **b)** The same problem with each data point expressed in a feature space with $f_1 = x_1 \times x_2$. Once transformed into this feature space, linear separation becomes straightforward.

terpreted as edge features, which in turn may be used to infer topological features, which may then be classified. As opposed to the XOR problem which has only two dimensions, input in the image classification problem may have millions of dimensions.

One of the challenges when creating features is determining the quality of a given feature set. In isolation, feature creation is an unsupervised learning problem and cannot be evaluated without additional information, such as an objective function or the performance of a second machine learning algorithm that uses the created features as input. For example, in clustering, a closely related unsupervised learning problem, the quality of clusters can either be determined by a measure, such as the standard deviation of elements within a cluster, or by the performance of a classification algorithm that uses the clusters as input. Data compression and prediction are other tasks that may be paired with feature creators.

The feature creation algorithm presented here is used in tandem with a reinforcement learning (RL) algorithm. Feature sets that enable the RL algorithm to collect more reward are considered superior, setting up a basis for comparison. Together the feature creator and RL algorithm form BECCA, a brain-emulating cognition and control architecture. (See Figure 2.) BECCA was designed to address the problem of natural world interaction—making intelligent agents capable of pursuing arbitrary goals in unfamiliar and complex environments.

1.1. Related work

The introduction to the 2003 NIPS Feature Extraction Challenge [10] gives an overview of common feature creation methods. [5] It differentiates feature selection from feature construction. In feature selection a few of the inputs are assumed to provide most of the information required for classification. For instance, of all the millions of tests and observations that could be made on a human body, only a few need to made to determine whether a patient has a cold: checking for sore throat, fever, and runny nose. A competent feature selection algorithm would be able to identify these discriminating features, given sufficient training data.

Alternatively, feature construction is useful when a combination of many inputs carries the discriminative information. In image classification, for example, a set of images of pepperoni pizza may not have any single pixels that have the same value in all images, so a feature selection approach would not work. Feature construction would be necessary, finding operations on the input that allow the category to be discriminated. Common methods include principal components analysis, support vector machines, linear discriminant analysis, Fourier transforms, and information theoretic tools. Entries to the Feature Extraction Challenge used both feature selection and feature construction approaches, and the winners are described in the same volume as the introduction. [17, e.g.]

It is interesting to note that, in a follow-up activity, organizers of the Feature Extraction Challenge tasked a class of undergraduates with creating their own entries. [6] Using relatively simple feature creation methods, the students improved upon the performance of the best challenge entries. This may have been due to the fact that the challenge consisted of two-class classification problems. More sophisticated tasks may require more complex feature sets. Whatever the cause, it demonstrates the room for growth that feature creation has as an academic field.

Another name for feature creation is "deep learning", [2] a label adopted by a research community seeking to discover and exploit the structure that underlies large, high-dimensional data. Deep learning approaches create high-level, low-dimensional feature representations using tools like Convolutional Neural Networks (CNNs) [8], Deep Belief Networks (DBNs) [7,3], and Deep SpatioTemporal Inference Network (DeSTIN) [1]. CNNs are designed to work with two-dimensional data, such as images, and they do not apply to arbitrarily structured data. By using several layers of Restricted Boltzmann Machines, DBNs are capable of generating sophisticated features that allow it to interpret novel inputs. However, they are typically applied to the supervised learning problem of discrimination, and require a substantial amount of labeled data in order to be adequately trained. Whether DBNs can be applied to the unsupervised learning problem of feature creation is unclear. DeSTIN incorporates both unsupervised and supervised learning methods and appears to be fully capable of hierarchical feature creation. It has been published only recently; future papers describing its operation and performance will allow it to be fully evaluated. Similar to these deep learning approaches, BECCA's feature creator is hierarchical, meaning that it repeatedly creates higher-level features from lower-level features. In contrast to them, it is incremental and on-line, efficiently incorporating new data as it is observed and updating itself after each time step.

2. Method

BECCA's feature creator takes in m inputs and produces n outputs, each on the interval $[0, 1]$, at every time step. It assumes nothing about the source of its inputs or the destination of its outputs. Inputs are treated the same whether they represent a pixel value, a sensed touch, or a joint position. The feature creator also takes in a scalar reward at each time step.

The feature creator first breaks the inputs into groups by finding sets of inputs that are, on average, somewhat correlated. Each group is a subspace of the full input space, and it is in these subspaces that features are defined. Whenever the inputs to a group are sufficiently different from the features in that groups, the inputs become a new feature and are added to the feature set. Also at each time step, the inputs are used to "vote" for the activity level of each feature by calculating their similarity to each. The activity of the most similar feature is preserved and the others are suppressed through a winner-take-all operation. The feature activities across all groups are concatenated to produce an output vector. A copy of the output vector is fed back and concatenated with the input vector as well, allowing the features to be constructed into yet higher-level features.

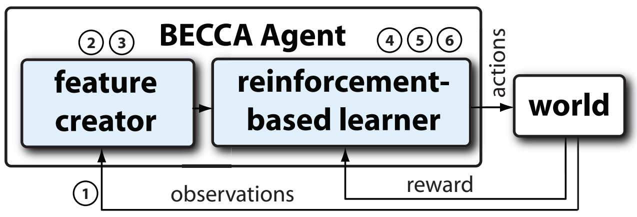

Figure 2. At each timestep, the BECCA agent completes one iteration of the sensing-learning-planning-acting loop, consisting of six major steps: 1) Reading in observations and reward. 2) Updating its feature set. 3) Expressing observations in terms of features. 4) Predicting likely outcomes based on an internal model. 5) Selecting an action based on the expected reward of likely outcomes. 6) Updating its world model.

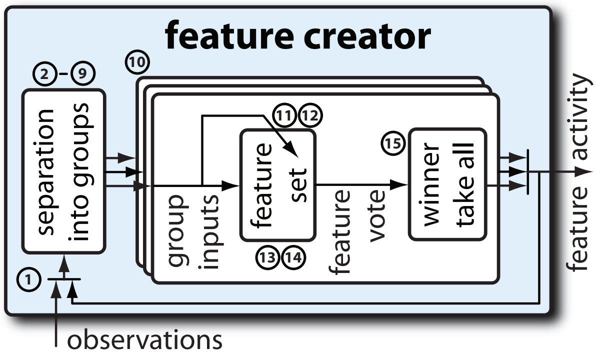

Figure 3. Block diagram of the feature creator, illustrating its operation. Numbered labels refer to steps in Algorithm 1.

The operation of the feature creator can be illustrated by providing it with a variety of inputs. Here, it was given two separate input sets, corresponding to two tasks, *watch* and *listen*. In the watch task, the inputs consisted of sections of images from the Caltech-256 data set. [4] These were converted to black and white and pixelized to a 10×10 array of super-pixels. A center-difference image was created by subtracting a portion of a pixel's neighbors' values from its own, and the 100-pixel array was normalized to fall on $[0, 1]$. The array, v, and its complement, $1 - v$, were passed to BECCA's feature creator, resulting in a 200-dimensional input vector.

In the listen task, an audio stream taken from the KUNM radio station in Albuquerque (89.9 FM) was recorded in 100 ms intervals. The frequency content of each snippet between 10 and 10,000 Hz was collapsed into bins of one-sixth of an octave (one whole note in modern music). The level of each bin was low-pass filtered and increases in bin level were used as inputs. The data that resulted from this pre-processing resembled a 60-channel graphic equalizer.

A block diagram describing the feature creator's operation (See Figure 3.) and pseudocode (See Table 2.) provide additional details. Full MATLAB (Mathworks, Natick, MA) code for BECCA and the watch and listen tasks is available as well. [13]

Algorithm 1 FEATURE CREATOR

Input: *observation* vector
Output: *feature_activity* vector

```
1:  form input vector by concatenating observation
        and previous feature_activity
2:  update estimate of correlation between inputs
3:  if MAX(correlation) $> C_1$ then
4:      add the two input elements achieving the
            maximum correlation to the new group
5:      while NOT(stop_condition_met) do
6:          find mean_correlation between each
                remaining input and group members
7:          if MAX(mean_correlation)$> C_2$ and
                group size $< C_3$ then
8:              add the input element to the group
9:          else stop_condition_met
10: for each group do
11:     if MIN (DISTANCE (input, features))$> C_4$ then
12:         add normalized input to set of features
13:     for each feature do
14:         feature_vote = feature $\cdot$ input
15:     feature_activity = WTA(feature_vote)
```

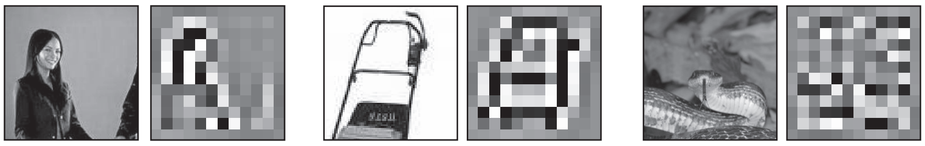

Figure 4. Examples of three inputs to the feature creator before and after preprocessing. From left to right the images are of a woman, a lawnmower, and a snake. (From [4].) When reduced to a 10×10 pixelation, the resulting image is unidentifiable, but retains coarse aspects of the original image. Light pixels represent locations that are brighter than their neighbors and dark pixels signify the opposite. Neutral gray pixels show locations that are approximately the same intensity as those around them.

3. Results

The feature creator stepped through 150,000 time steps in the watch task. Examples of the inputs at three different time steps are shown in Figure 4. During the course of learning, 264 groups were created, containing 1828 features in all. The complete set of features from one of the groups are in Figure 5, and samples of other groups can be seen in Figure 6. The entire set of features, broken out by group, can be found in an online supplement [15].

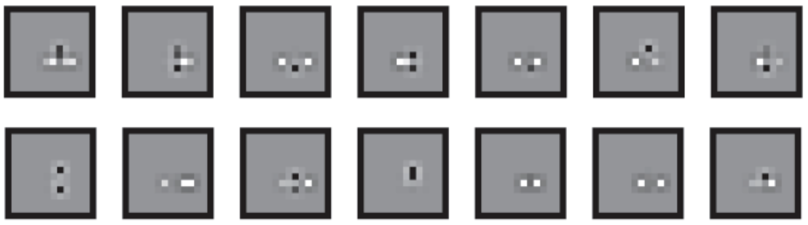

Figure 5. Receptive fields for all the features from one group. Although each feature is a linear combination of the same set of input pixels, the weighting of the pixels in each feature varies and is sometimes near zero. Inspection of the features suggests that they would be differentially activated by a wide variety of visual stimuli. Some would respond well to bright horizontal line segments and some to dark vertical line segments, some to continuous lines and some to line terminations.

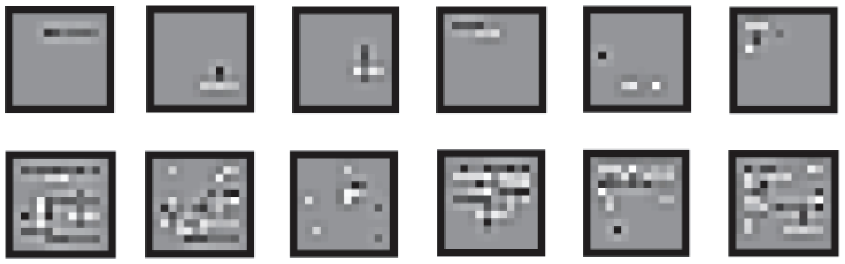

Figure 6. Receptive fields for features from 12 of the 264 groups. The six groups in the top row are first-level features, meaning that they are each composed of raw center-surround pixel inputs. The six groups on the bottom row are higher-level features. Each of them are composed of a mixture of first-level features and raw inputs. They tend to span a much larger portion of the visual field than first-level features.

In the listen task, the feature creator stepped through 15,000 time steps, creating 80 groups with 381 features. Examples of audio features can be seen in Figure 7. The entire feature set can be viewed in the online supplemental material [14]. A video showing both visual and audio representations of the features can be found here [12].

4. Discussion

BECCA's method of feature creation results in features that are sparse, that is, they are functions of only a small number of inputs. Sparsity is often considered a desirable attribute of a feature set. Principal Components Analysis and related methods lack it entirely, creating linear combinations of the entire input set. In order to encourage sparsity, many feature creation algorithms include a normalization term. Occasionally the l_0 norm is used, [18] (the count of the number of inputs used in a feature, the strongest sparsity condition) but more often it is the l_1 norm (a weaker sparsity condition, but more amenable to computation and analysis). By fixing the number of input elements in each group, BECCA's feature creator enforces a maximum l_0 norm, creating strongly sparse features.

By not making assumptions about the source of its inputs, BECCA's feature creator gains the capability of handling inputs of all modalities, and of combining them. It is not uncommon for feature creators to assume that incoming data is from a one-dimensional

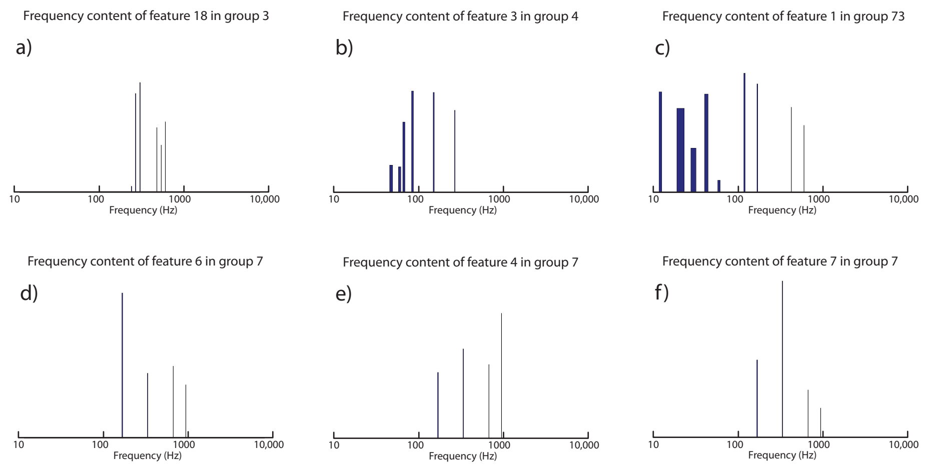

Figure 7. Receptive fields for six features from the listen task. An octave, a factor of two difference in frequency, is represented by a fixed interval on these logarithmically-scaled plots. The harmonic overtones of a fundamental frequency are its integer multiples. The magnitudes of the various overtones determine the quality or timbre of the sound. The difference between, say, a violin and a flute playing the same note lies primarily in their differing patterns of overtones. Due to resonant nature of sounds, single sources often produce sounds consisting of a fundamental frequence with its harmonic overtones. These commonly occurring patterns are observed in the groups and features created. **a)** This feature shows a fundamental frequency cluster and its first harmonic, separated by an interval of one octave. **b)** In this feature, a fundamental frequency of approximately 45 Hz is accompanied by its first, third, and seventh harmonics (two, four, and eight times the fundamental frequency, respectively) and several lower non-harmonic overtones. **c)** Where both a) and b) are composed of raw inputs, this feature combines other features to make a more complex, high-level audio feature. There are many examples of high-level features like this one in the created feature set. **d-f)** These features are all from the same group. They illustrate receptive fields of the same fundamental frequency, but with different timbres. In each case, the fundamental frequency is accompanied by its first, third, and fifth overtones. Changes in their relative amplitudes make them sensitive to sounds with distinctly different qualities. (See [12] for a full audiovisual representation.)

array of sensors, as in an encoder, or a two-dimensional sensor array, as in a camera. Hand-crafted kernels or transformations are often built in. These engineered approaches can produce striking results, but at the cost of some generality. The approach presented here aims to be as general as possible, allowing a programmer to be agnostic about the source and meaning of inputs. This opens the door for automatic feature creation in not only vision and hearing, demonstrated here, but also in laser scanning, whisker contact, radiation detection, network traffic monitoring, and any other conceivable sensor modality, in addition to combinations of all the above.

Feature creation allows robots to interact with their environments in increasingly sophisticated ways. In cases where their environments and tasks are unfamiliar, feature creation may be necessary for any better-than-random performance on a task. It is a means of abstracting specific experience into a more symbolic representation. In a sufficiently complex environment, an agent is unlikely to experience the exact same set of inputs more than once. If that agent is to learn from its experience, it must have a way to relate its current experience to previous ones. Abstract features make this possible. They allow one to claim that two objects are both books, even though they are of different sizes, weights, colors, ages, smells, and textures. And in this way, they allow experience gained with one book to be transferred to the other. Natural language acquisition can be understood as feature creation operating at an advanced level. The symbols for words can be created from their corresponding printed and spoken forms and the visual and other sensory experiences associated with them. Automatic feature creation will almost certainly be a capability of any machine capable of human-level learning behavior and performance.

BECCA's feature creator was inspired by evidence of biological feature creation. At a high level, human capabilities of analogy, association, generalization, and abstraction suggest that arbitrary symbols may be combined and related to one another. Our ability to learn and understand such things as microcircuit design and quantum physics suggests that these symbols are acquired rather than evolved and innate. The mammalian primary visual area of the cerebral cortex, V1, has inputs that are excited by small center-surround stimulus fields and outputs that are activated by appropriately oriented line segments. Other areas of the cortex, such as V2, V4, and the medio-temporal area, take these outputs as their inputs and build them into outputs that respond to increasingly complex features. Other areas of the cortex appear to serve the same function for touch [11] and hearing, as demonstrated by the aural representation in bats that enables echolocation. Re-routing sensory inputs to a new cortical location results in similar feature creation phenomena, [16] hinting that the cortex may be performing a similar function throughout its extent. If that is the case, the function it performs may be nothing more than feature creation. However, there is still far too little data available to certify or disprove such a statement. It is at most a conjecture, and regardless of its accuracy, the feature creation observed in the cortex provides a rich source of insights for the development of machine learning algorithms seeking to do the same.

Acknowledgements

This work was supported by the Laboratory Directed Research and Development program at Sandia National Laboratories. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

References

[1] I. Arel, D. Rose, and B. Coop. DeSTIN: A deep learning architecture with application to high-dimensional robust pattern recognition. In *Proc. AAAI Workshop Biologically Inspired Cognitive Architectures (BICA)*, 2009.

[2] I. Arel, D. C. Rose, and T. P. Karnowski. Deep machine learning—a new frontier in artificial intelligence research. *IEEE Computational Intelligence Magazine*, November 2010.

[3] I. Fasel and J. Berry. Deep belief networks for real-time extraction of tongue contours from ultrasound during speech. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2010.

[4] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007. http://authors.library.caltech.edu/7694.

[5] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[6] I. Guyon, J. Li, T. Mader, P. A. Pletscher, G. Schneider, and M. Uhr. Competitive baseline methods set new standards for the nips 2003 feature selection benchmark. *Pattern Recognition Letters*, 28:1438–1444, 2007.

[7] G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527Ű1554, 2006.

[8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.

[9] M. Minsky and S. Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, 1969.

[10] Neural Information Processing Systems. NIPS Feature Extraction Challenge, 2003. http://www.nipsfsc.ecs.soton.ac.uk/, Accessed April 25, 2011.

[11] J. R. Phillips, K. O. Johnson, and S. S. Hsiao. Spatial pattern representation and transformation in monkey somatosensory cortex. *Proceedings of the National Academy of Sciences*, 85:1317–1321, 1988.

[12] B. Rohrer. Automatically created audio features. http://www.sandia.gov/~brrohre/video/AudioFeatures.mp4, 2011.

[13] B. Rohrer. BECCA code page. http://www.sandia.gov/~brrohre/code.html, 2011.

[14] B. Rohrer. Listen task features. http://www.sandia.gov/~brrohre/doc/Rohrer11BicaSuppB.pdf, 2011.

[15] B. Rohrer. Watch task features. http://www.sandia.gov/~brrohre/doc/Rohrer11BicaSuppA.pdf, 2011.

[16] J. Sharma, A. Angelucci, and M. Sur. Induction of visual orientation modules in auditory cortex. *Nature*, 404:841–847, 2000.

[17] K. Torkkola. Feature extraction by non-parametric mutual information maximization. *Journal of Machine Learning Research*, 3:1415–1438, 2003.

[18] J. Weston, A. Elisseff, B. Schoelkopf, and M. Tipping. Use of the zero norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.