Performance Improvement by Input Adaptation Using Modified Hebbian Learning

Yong-Jian Zheng and Bir Bhanu
College of Engineering
University of California
Riverside, CA 92521

Abstract

We address the issue of how to improve the performance of an image understanding system which consists of a set of algorithms. We discuss the problem of characterization of the input images/data to a system and its performance. We present two general methodologies for performance improvement. They are based on optimization of parameters of algorithms and adaptation of the input to an algorithm. We focus on the second methodology and use modified Hebbian learning rules to build adaptive feature extractors which transform the input data into the desired form for a given algorithm. We demonstrate the feasibility of the approach by designing an input adaptor for target detection and showing results using optical, SAR, and FLIR images.

1 Introduction

This paper is motivated by the increased demand for new theories and methodologies to characterize and to improve system performance [3, 4, 5, 7, 6] and to minimize the effort needed for the development of robust image understanding systems for practical real-world applications such as ATD/R. The key idea of the paper is that the performance of a given algorithm can be improved by adding an adaptor between the input data and the algorithm. This is an input adaptive process and is based on the observation that most algorithms would perform well if the desired input data can be provided to them. Different kinds of Hebbian-like learning rules are introduced and applied to developing such adaptors. The reasons why this contribution is considered important are:

- Performance improvement is a real challenge, from both theoretical and practical points of view. It deals with many unsolved problems and issues such as how to characterize the input data, how to characterize the performance of a given system, what are the factors which affect the performance of a system, what are the ways and methodologies to improve the performance of a system, and how to realize these systems.
- To characterize the input images/data, a perturbation model based method is introduced. It divides the input into three groups labeled as perfect, expected, and unexpected. Based on this grouping, it is possible to define measures, such as system sensitivity, system controllability, system dynamic range, and system reliability, to characterize an algorithm and a system.
- Two general methodologies are introduced for improving the performance of a given algorithm and they are based on optimization of parameters of an algorithm and adaptation of the input to an algorithm. Comparison shows that methodology based on adapting the input has some inherent advantages over the parameter optimizing based approach and it is suitable for many real-world applications. This approach is presented in detail and experimental results are given to support the idea.

There is no work by others that is directly or closely related to our approach. However, some of the issues that we discuss are treated by others from different points of view. Haralick [4] discusses the meaning of performance characterization in general and introduces the protocols which can be used to characterize the performance of an algorithm. Chen [2] gives a good review of neural network architectures for feature extraction and dimension reduction.

2 Characterizing Computer Vision Systems

2.1 Characterizing Images and Input Data

A system used in a real-world vision application usually consists of some subsystems, each of which just accomplishes a specific task. The performance of the complete system is possible if each of the subsystems can be characterized [4].

A subsystem for visual information processing, as shown in Figure 1, is composed of an algorithm \( F \), input \( X \), parameter \( \Omega \), and output \( Y \). For each \( X \) and \( \Omega \), a \( Y \) is produced by an algorithm \( F \).

Now the question is why the output produced by an algorithm contains variations, imperfections, or sometimes even errors. The answer to this question is the
It is clear that this classification of the input images or data is algorithm-dependent. A given algorithm has its own definition about classes of the input images or data. Unfortunately, most commonly used algorithms provide no information about the classification of their input image or data.

2.2 Framework for System Characterization

As just mentioned, characterizing a computer vision system means characterizing its subsystems and algorithms. An algorithm $F$, as shown in Figure 1, can be thought of as a set of functions and each function $f \in F$ describes the dependence of a given element $Y \in Y$ on $X$ and $\Omega$. Because of the relationship $Y = f(X, \Omega)$, the behavior of $Y$ depends on the property of $X$ and $\Omega$:

- If $X$ is ideal and $\Omega$ is tuned perfectly, $Y$ produced by $f$ should also be perfect. This means that the algorithm requires the ideal $X$ and perfect $\Omega$ in order to provide the ideal $Y$.
- If $X$ is imperfect and contains some random variations, $Y$ produced by $f$ is imperfect and should also have some random components. This means that there is a correspondence of the random variations and imperfections which the algorithm produces on the output caused by the random variations and imperfections on the input. The output variations and imperfections of $f$ can be sometimes reduced by tuning $\Omega$, if it has good gradients with respect to $\Omega$.
- If $X$ is unexpected because its variations are not modeled, the behavior of $f$ depends on its property. If $f$ is robust, the unexpected $X$ can be detected and handled carefully so that no unreasonable $Y$ is produced by $f$. Otherwise, if $f$ is not robust, $Y$ produced by $f$ is unexpected.

To show this, the four synthetic images just generated are used as the input data to two popular computer vision algorithms, namely edge finding and histogram based image thresholding. Figure 3 and Figure 4 show the performance of these two algorithms. It can be seen that both algorithms perform well if the input image is ideal or just contains a normally distributed random perturbation. This is because this perturbation is considered and modeled in both algorithms. Actually each algorithm has its own perturbation model and can, therefore, deal with some input perturbations. However, if the input perturbation is unexpected and does not fulfill its model in the algorithm, the algorithm cannot produce reasonable results (see the results for case c and d in Figure 3 and the results for case d in Figure 4).

Based on the above discussion, a computer vision sub-system can be characterized by using the following definitions:

- **System Sensitivity** describes the grade of the output variations of the system if the input data are expected and the parameters are tuned properly.
- **System Controllability** describes the grade of the output variations caused only by tuning the param-
method for problem-solving is to develop a new algorithm or system which is more sophisticated and contains appropriate perturbation models of the input data. This, of course, leads to the fact that the newly developed algorithm or system becomes quite complicated and time consuming to account for large expected/unexpected variation in the input data. To deal with this dilemma the adaptability should be added to the available algorithms. Generally speaking, there are two methodologies for adaptation.

The first methodology is based on the consideration that some algorithms and systems have certain controllability and their performance can be improved by tuning their parameters [1, 2, 3]. To find the best parameter set for the given input data a learning and optimizing process is usually required. This methodology is, therefore, parameter optimizing oriented. As shown in Figure 5, parameter optimizing based methodology employs different parameter set for different input data in order to obtain the optimal output. However, this methodology suffers from some inherent shortcomings:

- It is driven by both the input data and the output data. It has to have an off-line learning phase. This means that the correspondence between the input data and the appropriate parameter sets should be first established during an off-line learning process before the algorithm can possess the required adaptability. To perform the off-line learning requires good and enough data samples. The data samples should cover input situations as far as possible and their number may be too large if the number of input situations increases. This leads to the difficulty of sample collection because some input situations are not predictable. Besides, the off-line learning process is usually time consuming.

- In order to use the trained algorithm, information about the possible category of the input data is needed before the appropriate parameter set can be switched on. This means that the trained algorithm works only with an additional input identifier which triggers the corresponding parameter set. Certainly the design of such an identifier is as hard as that of the algorithm itself.

- The performance of an algorithm cannot be always improved by optimizing the parameter set because the gradients of objective functions of some algorithms with respect to their parameters are too small. So not all algorithms can be improved by using this methodology.

Due to these shortcomings, the potential application of this off-line learning and parameter optimizing based methodology to a real-world problem is limited, although some research has been recently done in this area [1].

The second methodology for performance improvement is based on the observation that most algorithms would perform well if their input data are “friendly”, as discussed above. Thus, the performance of almost all commonly used algorithms can be improved by adding an adaptor between the input data and the algorithm (see Figure 6). An ideal adaptor should automatically judge the input data, provide the desired input data to an algorithm, and learn something from this process in order

2.3 Parameter Optimization Versus Input Adaptation

So far the problem of why a computer vision algorithm produces variations and imperfections on its output and how to characterize the performance of a given algorithm have been treated. All of these are related to the ultimate aim of finding some ways to improve the performance of computer vision algorithms and systems.

If the performance of an algorithm or system cannot satisfy the needs of a real-world application, the most used...
are linked by algorithms which transform an input representation to an output representation. As mentioned above, many commonly used algorithms are only designed to deal with those input representations which are relatively "friendly". This means that these input representations contain less spurious and erroneous information and have more explicitness. To distinguish these desired representations from others, we define them as salient features. So salient features are those input representations which possess some “nice” properties, and therefore, desired by a given algorithm.

Most commonly used algorithms can show good performance only if their input representations have some "friendly" characteristics. To keep their performance high even if the input representations are not so "friendly", adaptors are needed which transform the input representations to some salient features. Thus, an adaptor can also be regarded as a salient feature extractor. The key issue in input adapting methodology for performance improvement is how to design an adaptor or salient feature extractor for each algorithm at each stage of the representation transformation.

3.2 Optimal Feature Extraction

From a mathematical viewpoint, feature extraction is a transformation from an m-dimensional input representation x to a n-dimensional output representation v, so that n \leq m and for each v \in v the expected value of \rho(v) is minimized:

\[ E(\rho(v)) = \int_{-\infty}^{+\infty} \rho(v)p(v)dv \to \min, \]

where \rho(\cdot) is a “loss” function, \( E(\cdot) \) is the risk (expected value of the loss), and \( p(\cdot) \) is the probability density function of \( v \). This means that the transformed representation \( v \) should be less redundant (because of \( n \leq m \)) and salient (because of \( E(\rho(v)) \to \min, v \in v \)). Thus, \( E(\cdot) \) is a measure of saliency which depends on the loss function \( \rho(\cdot) \).

A simple example of the representation transformation is the linear mapping \( W \) which transforms the m-dimensional input representation x to n-dimensional output representation v by using

\[ v = Wx = (w_1, w_2, \ldots, w_m)^T x. \]

In this case, \( W \) is a feature extractor if \( v \) has some nice properties. The feature extractor \( W \) can be realized by using a single-layer linear feed-forward network. As can be seen, the basic unit of this network is a m to 1 mapping

\[ v = w^T x = x^T w, \quad v \in v. \]

The basic unit can also be nonlinear. In this case the m to 1 mapping is formulated by

\[ v = \tau(w^T x) = \tau(x^T w), \quad v \in v, \]

where \( \tau(\cdot) \) is nonlinear function. The mapping (3) or (4) is salient or interesting if \( E(\rho(v)) \) is minimized. The key issue of constructing a salient feature extractor is thus the design of the loss function.

Before the loss function can be designed, the question of which \( v \) is “salient” or “interesting” should be first
Figure 7: A point cloud in a 2-dimensional space.

Table 1: Different learning rules

<table>
<thead>
<tr>
<th>Type</th>
<th>The Learning Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>$\Delta w_i = \eta x_i$</td>
</tr>
<tr>
<td>II</td>
<td>$\Delta w_i = \eta (x_i - u w_i)$</td>
</tr>
<tr>
<td>III</td>
<td>$\Delta w_i = \eta \left[ v^2 - E(v^2) v \right] x_i$</td>
</tr>
<tr>
<td>IV</td>
<td>$\Delta w_i = \eta \left[ v^2 - E(v^2) v \right] (x_i - u w_i)$</td>
</tr>
</tbody>
</table>

defined. Certainly, no universal agreement on this question can be expected. Two general definitions about the saliency of $v$ that we have are:

- **Expressiveness**: $v$ is salient if it is **expressive**.
- **Discrimination**: $v$ is salient if it is **discriminating**.

In the following, the problem of how to extract these features is addressed.

### 3.3 Expressive Versus Discriminating

Let us consider a set of $m$-dimensional input representations $\mathcal{X} = \{x_1, x_2, \ldots, x_i\}$ which builds a "cloud" of points in the $m$-dimensional space. It is clear that each point $x \in \mathcal{X}$ can be projected onto a direction determined by the vector $w$ by using Equation (3) or (4) and the result of this projection is $v$. Figure 7 just shows a case of $m = 2$. Now the problem is which projection direction is interesting.

As shown in Figure 7 the first interesting direction is $v_E$ because the projections of all points onto this direction have the maximal variance and $v_E$ is, therefore, **expressive**. It can be proved that $v_E$ is determined by that $w$ which is the largest eigenvector associated with the largest eigenvalue of the correlation matrix

$$Q = E(xx^T).$$

The direction $v_E$ allows faithful representation of the input data and the projection of the point cloud onto this direction can also show interesting structure if the cloud contains a few clusters and the separation between clusters is larger than the internal scatter of the clusters.

However, the direction $v_E$ can lead us astray if the cloud shows too many isotropically distributed clusters or if there are meaningless variables with a high noise level. In these two cases, the output representation $v_E$ doesn't allow discrimination between clusters (see the example in Figure 7).

The second interesting direction in Figure 7 is $v_D$ because the projections of all points onto this direction can enable us to better distinguish the interesting structure (clusters) presented in the cloud and $v_D$ is, therefore, **discriminating**.

To develop learning systems for adaptive extraction of expressive or discriminating features from the input data, four different learning rules are used (see Table 1). They are based on four different loss functions as shown in Table 2. While the learning rule I and II are suitable for extracting expressive features, the learning rule III and IV can be used for extracting discriminating features. It is clear that the rule I is famous **plain Hebbian learning** rule and the rules II, III, and IV can be thought of as modified Hebbian learning rules (12).

### 4 Experimental Results

In this section some initial experimental results are presented to show the feasibility of input adapting based approaches for performance improvement of computer vision systems.

#### 4.1 Input Adaption for Image Thresholding Algorithm

According to Figure 6, input adapting approaches for performance improvement try to design an adaptor for each algorithm. The adaptor should transform the input data, whose perturbations may not be modeled or considered by a given algorithm, so that the transformed data have a form which is desired by the algorithm. It is expected that with an adaptor the image thresholding algorithm (see Figure 4), for instance, will perform better even if the input data are unexpected. This means that with this adaptor the dynamic range of the thresholding algorithm should be enlarged and its robustness should also be increased even if the unexpected input data are presented.

Figure 8 shows an adaptor which is designed for the image thresholding algorithm. The key idea for designing this adaptor is to decompose the input image into
some local measure images and then to adaptively extract salient features from these local measure images based on the modified Hebbian learning rules presented above. In order to derive local measures for each pixel in the input image, the quadrature Gabor filter kernels

\[ G_+(\omega, \phi) = \exp \left[ \frac{-\lambda^2 \omega^2 (x^2 + y^2)}{4\pi} \right] \cos[\omega(x \cos \phi + y \sin \phi)] \] (5)

\[ G_-(\omega, \phi) = \exp \left[ \frac{-\lambda^2 \omega^2 (x^2 + y^2)}{4\pi} \right] \sin[\omega(x \cos \phi + y \sin \phi)] \] (6)

are applied to decompose the input image \( I(x, y) \) by using

\[ I_+(x, y, \omega, \phi) = G_+(\omega, \phi) \ast I(x, y), \] (7)

\[ I_-(x, y, \omega, \phi) = G_-(\omega, \phi) \ast I(x, y), \] (8)

where \( \omega \) and \( \phi \) are the modulation (center) frequency and orientation, respectively, of the Gabor filter kernel; \( \lambda \) is the ratio of the channel bandwidth and the modulation frequency; and \( I_+(x, y, \omega, \phi) \) and \( I_-(x, y, \omega, \phi) \) are Gabor space image descriptions. From these descriptions it is easy to derive some local measures. In Figure 8 the power

\[ p(x, y, \omega, \phi) = I_+^2(x, y, \omega, \phi) + I_-^2(x, y, \omega, \phi) \] (9)

is used as a local measure of the input image \( I(x, y) \). So far power images can be obtained and \( m \) depends on the quantization of \( \omega \) and \( \phi \). This means a local measure vector with \( m \) elements is associated with each pixel of the input image.

Each element of the local measure vector is a representation to describe the local property of the input image but it is not just the right feature to discriminate clusters depicted in the input image. The most discriminating feature should be found in the \( m \)-dimensional local measure space based on the structure presented by all local measure vectors in the input image. This requires a \( m \) to 1 feature extractor \( A \) which is trained by using the learning rule III or IV as listed in Table 1.

To reduce high frequency components in the input data two feature extractors \( B^* \) and \( C^* \) are introduced into the adaptor shown in Figure 8. They are actually two convolution kernels with \( n \times n \) elements which should be trained by using the learning rule I or II as listed in Table 1.

After the convolution using \( B^* \) and \( C^* \) two feature images can be produced which should be integrated by the feature extractor \( D \) in order to supply desired images for the thresholding algorithm. The feature extractor \( D \) performs a 2 to 1 transformation and is trained by using the learning rule III or IV as listed in Table 1.

Now the adaptor is able to produce desired input images for the thresholding algorithm (see Figure 8). It is obvious that the output images in Figure 8 are more "friendly" than the input images in Figure 8 to be used as the input images for the thresholding algorithm because the object in the center is better discriminated from the background. Thus, the performance of the thresholding algorithm can be improved by using the adaptor (see

Figure 8: Input adapting for the image thresholding algorithm.

Figure 9: Improving the Performance of Thresholding Algorithm.

Figure 9). In Figure 9, the object in the center is better discriminated from the background even if the input image d, which is not desired by the thresholding algorithm (see Figure 4), is presented.

4.2 Target Detection Using Multisensor Data

As mentioned above, ATD/R is an important application of computer vision technology. The first step of ATD/R process is the detection of potential targets in the input data. The performance of target detection affects the whole target recognition process. So the emphasis here is on how to enable a already-developed target detection algorithm to deal with different kinds of sensor data such as, optical, infrared, and SAR images without developing a totally new algorithm for each sensor.

Figure 9 shows that a target detection system made up of a simple thresholding algorithm plus an input adaptor is able to handle the synthetic input data with different degrees of clutter. It is thus desired to know if this input adapting approach also works in the presence of real sensor data.

Figure 10 shows the test result of target detection system using SAR image data. The column a shows the input images. The column b shows the test results using the thresholding algorithm. The column c shows the test results using the thresholding algorithm plus the input adaptor. It can be seen that even a simple algorithm
can perform well if its input data are properly prepared by an input adaptor. This means that adding an input adaptor can enlarge the dynamic range of an algorithm and improve its performance.

Figure 11 shows another example of target detection in a FLIR (Forward Looking Infrared) image by using the same system. Again, the image a is the input image. The image b and c show the test results using the thresholding algorithm without and with the input adaptor. As can be seen, the performance of the system is satisfied even when the input image has different properties as used before.

5 Conclusions

In this paper the attention was paid on how to characterize the input images/data, how to characterize computer vision systems at algorithm level, what are general paradigms to improve the performance of a system, and how to make ready-made available algorithms perform well without changing their internal structure. The answer to these questions is the basis to gain a further understanding of the problem. As compared with the parameter optimizing based methodology for performance improvement, the input adapting methodology is unsupervised in nature and it is driven by the input data. This is desired by some applications such as ATD/R in a changing environment. In summary, the input adapt-

Acknowledgment

The help of Yeli Chen in preparing this paper is gratefully acknowledged.

References

IMAGE UNDERSTANDING WORKSHOP

Proceedings of a Workshop
held in
Palm Springs, California

February 12 - 15, 1996

Volume II

Sponsored by:
Advanced Research Projects Agency
Information Systems Office

This document contains copies of reports prepared for the ARPA Image Understanding Workshop. Included are Principal Investigator reports and technical results from both the basic and strategic computing programs within ARPA/ISO sponsored projects and certain technical reports from selected scientists from other organizations.

APPROVED FOR PUBLIC RELEASE
DISTRIBUTION UNLIMITED

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Advanced Research Projects Agency or the United States Government.