Content-Based Image Retrieval Using Hierarchical Temporal Memory

Bruce Bobier

bbobier@uoguelph.ca, Department of Computing and Information Science, University of Guelph, Guelph, ON, N1G 2W1;

ABSTRACT

Tens of thousands of architectural elevation drawings are available online, yet current text- and content-based approaches have had limited success with indexing these drawings, and with retrieving them via usable and intuitive interface that is robust to noise, occlusions and affine transformations. In this paper, the use of color, shape and spatial layout features are surveyed, as are several querying interfaces for content-based image retrieval (CBIR). Due to the limitations of the feature-based approaches, a new CBIR system is introduced that uses Numenta's NuPic implementation of Hierarchical Temporal Memory for automatic indexing and a sketch-based and iconic index querying interface. Experimentation shows the system is robust for recognizing query images under varying amounts of noise, distortion, occlusion, and rotation.

Keywords: Content-based Image Retrieval, Hierarchical Temporal Memory, Querying Interfaces

1. INTRODUCTION

The vast amount of digital visual information stored in online databases has increased the need for more efficient and effective means of its management and retrieval. Content-based image retrieval (CBIR) refers to the problem of searching for images in large databases using techniques from computer vision, artificial intelligence and pattern recognition.[4] Content-based approaches are needed for the management of visual information in cases where textual annotations for images are either incomplete or nonexistent. By combining textual annotations with additional image features, CBIR may offer the ability to improve retrieval precision and provide better understanding and representation of the stored information.

CBIR has been used on numerous types of visual information including photographs, medical images, videos, line drawings, sketches, artwork, and 3D images.[39] In this article, the focus is placed on CBIR in the context of architectural elevation drawings. Formally, an elevation drawing is a representation of a buildings geometrical exterior as perceived from a horizontal viewpoint, without dimensional perspective. The drawings are manually created with pen and paper and digitized as binary, grey scale or color images using a scanner or overhead camera. Presently, thousands of these drawings are stored in online databases, with even more known to exist in offline archives and collections. In the coming years when the offline collections are digitized and combined with online collections, the need for an effective means of managing and retrieving these images based on their contents will be of even greater importance.

As underlined by Gross and Do,[16] the current indexing scheme for images often relies on the name, date, location, and architect of the represented building, which results in databases that index the circumstances of the image rather than the content. Color and texture-based indexing and retrieval is similarly limited, as for most architectural design tasks, color and shading features are less relevant to the user than are form and spatial features.

The functioning of most CBIR systems can be divided into two main steps. In the first step, for each image in the database, a feature vector is computed from some of its image properties such as color, texture, shape, and spatial layout, and the feature vector is then indexed in a feature database. In the second step, a query is submitted, for which a feature vector is computed and compared to other feature vectors in the feature database. From this comparison, the images that are most similar to the query are retrieved and displayed to the user.

The earliest attempts at CBIR utilized computer vision algorithms that performed feature based similarity searches on image collections.[6,24] Subsequent research efforts such as Webseek[34] and WebSeer[8] employed similarity based searches in online search engines. More recently, research has focused on CBIR systems that are

able to understand the images' semantic information, and are able to represent their content in a manner that is intuitive to humans. Examples of such systems include the minimum probability of error system[37] and the multiple Bernoulli relevance model.[5] Feature based CBIR systems are often non-intuitive and difficult to use, and the CBIR system presented in this article aims to understand a query's semantics, rather than the low-level image features. This paper presents a new approach to CBIR, dubbed HTMCBIR, that uses the Numenta NuPic platform[13] to provide an intelligent system for indexing and retrieval using a sketch-based interface.

The remainder of this article is organized as follows. Section 2 discusses the use of color, shape and spatial layout as indexing features. Section 3 reviews approaches to querying interfaces and Section 4 introduces Hierarchical Temporal Memory. Section 5 introduces the HTMCBIR application, and Section 6 discusses the experimental results of the system. Section 7 concludes with a summary of the contributions of HTMCBIR, and Section 8 and discusses future work for the HTMCBIR system. Finally, Section 9 summarizes this article.

2. FEATURES

In this section, the use of color, shape and spatial layout features are reviewed. Other features, such as texture and combinations of multiple features are described in the literature (See[7] for a recent survey), but are consciously omitted from this article as they are not applicable to most elevation drawings. In a study about user cognitive processes during image retrieval, Schomaker et al.[32] found that users are more interested in retrieving images based on their shape than their color and texture. Considering this, and that most elevation drawings are grey scale or binary images, the main focus of this section is on shape and spatial layout features.

2.1. Color Features

Color features are the most commonly used descriptors for image indexing, as they may be easily extracted directly from an image's pixel intensities.[39] For indexing using color features, given a query, the problem is to retrieve all images that have a color composition similar to that of the query. Color features may be obtained in numerous ways, such as by calculating a color histogram across an entire image, over a subimage, or over a segmented area. Most commonly, color content is characterized by a color histogram, which is a multidimensional histogram of the color distribution in an image.[39] The similarity of a target image's histogram may be quickly calculated by comparing it to that of a query image's using a histogram intersection distance measure.[33,35] Additional distance measures have been proposed for color indexing. The quadratic distance proposed by Hafner et al.[17] aims to capture the perceptual similarity of two colors and performs similarity matching by representing colors using color histogram bins. Huang et al.[21] extended this work by introducing a new type of color feature, the color correlogram, which characterizes how the spatial correlation of color pairs changes with spatial distance.The color feature introduced by Gevers and Smeulder[14] was shown to be a more accurate approach to color histogram matching, as their color models are invariant to illumination, object geometry and viewing position. Color moments may also be used as a statistical measurement to characterize the color histogram distributions, where the first, second and third moments of the histogram describe the average, variance and skewness of the color histogram respectively. However, as the majority of elevation drawings are grey scale or binary images, the use of color features is largely not applicable. The use of grey scale features may address this problem, although such an approach fails to provide a means of capturing any of the semantic information contained by the image.

2.2. Shape

Using shape features for indexing is a less developed and significantly more complex problem than is color. This problem is complex because in order to describe an object's shape features in an image, the object must first be found. Locating an object can be performed using numerous segmentation techniques that typically rely on the combination of color and textural information with region-growing algorithms.[15] The problem of accurately segmenting an object by strictly using pixel information is in turn complexified by changes in the object shape that result from it being perceived from different view points, the complexity of the object itself, and the presence of occlusions, shadows, and additional noise from preprocessing.[4] Further, it is generally expected that an object's shape description is invariant to affine transformations, which is not supported by all approaches.

Generally, approaches to indexing using shape features are divided into two categories: global image approaches and global object approaches. With global image approaches, the goal is to transform the entire image's color information from the spatial domain to color variation information in the frequency domain.[38] Rather than attempting to characterize actual shape information, these approaches encode the color and intensity transitions of an image, which often occur at object boundaries.

Wavelet-based approaches are an example of global image approaches that transform the 2D color information into different frequency components. These components represent the original image as a linear combination of wavelet functions, and thereby allow the image to be indexed by processing its wavelet coefficients.[38] The set of wavelet coefficients may also be used as a feature vector for subsequent matching by computing the set of wavelet coefficients from the query image and performing a comparison.

By encoding the entire image, including irrelevant background information unrelated to the objects contained by the image, global image approaches are inherently limited in their ability to accurately index images and interpret queries. Further, as such approaches do not retain the actual shape information, it is not possible for a similarity measure between the query and target image to be computed. Finally, the nature of global transforms cannot be used to match a query shape with an object occupying only part of the target image, making their application limited to specific problem domains.

Global object approaches on the other hand, operate on the local features extracted from a complete object or contour itself. Presently, most CBIR systems using global object methods operate on two classes of 2D shape features: contour-based and region-based, or a combination of the two. Contour-based shape features describe the outer boundary or closed curve that outlines the object. This boundary can be represented by numerous descriptors, such as Fourier coefficients,[2] polygonal approximations,[30] autoregressive models[22] and splines.[27] The use of contour-based features (e.g. polygons or splines) enables a sketch-based query interface to be easily implemented, as the query sketch can be deformed to match the shape of the target model (i.e. the representation of the stored image objects), where the amount of deformation required serves as a similarity metric.[4] Using the deformable template approach, the amount of permissible shape variability is equivalent to that of the maximal deformation from a query image to a target model.

Region-based shape features differ from contour-based features by describing an object's inner area within a closed boundary. Note that both descriptions may be easily interchanged, as one may serve as the basis to compute the other (i.e. filling in the region or tracing its boundary). Region-based features can also be specified by numerous descriptors including skeletons,[23] point sets,[41] Zernike moments,[26] and moment invariants.[20,43] For line drawings, skeletons are among the most common shape representation. The skeleton of a given object is produced by iteratively thinning the object's edges until only a unit wide axis of symmetry between each edge's boundary is preserved. The resulting skeleton may then be compactly represented as a graph.

A caveat of global object approaches is that they require the objects in the image to be properly segmented beforehand, which is a complex problem on its own. Further, global object approaches are typically not robust to occlusions and noise.[38]

2.3. Spatial Layout

Spatial layout is a newer and more powerful indexing feature for describing content that extends the principles of shape features to characterize the spatial relationship of objects in terms of their topological and directional organization.[40] Topological relationships specify the spatial relationship between the boundaries of objects (e.g. "contained by", "beside"), whereas directional relationships describe the relative positioning of objects (e.g. "above", "on the left of"). Since these relationships are typically symbolically represented by a tree or graph, a similarity measure of two images can be computed by comparing their corresponding graph representations. One commonly used graph is an attributed relational graphs, where each graph node represents an object and each edge represents a relationship. However, as graph isomorphism is an NP problem and subgraph isomorphism is NP-complete,[7] comparing a query graph with a set of graphs, or finding instances of a subgraph in a large database of graphs becomes impractical.

3. QUERYING INTERFACES

The user interface in CBIR systems typically consists of a query specification mechanism and a means of presenting the results. Formulating and specifying a query can be accomplished using several approaches, the most common of which is to describe the desired image using textual keywords, which requires that the image data has been previously annotated. Keyword searches are also limited by their low scalability, ability to represent visual information in very narrow contexts and that users prefer to retrieve images based on their content, rather than their associated keywords.[28]

Iconic indexes have also been used for querying, and take the form of symbolic descriptors of image data or relationships. Iconic indexes may also include actual values for object features or utilize abstract images that represent the salient features of the original image.[36] The use of iconic indexes is closely tied with the query-by-visual-example approach, which attempts to reduce the user's cognitive load by exploiting their innate capabilities for image analysis and interpretation.[29] Iconic indexes are among the more suitable approaches to querying elevation drawing databases, as the hierarchical organization of architectural elements enables the user to iteratively refine their query using multiple iconic indexes (e.g. window - transom - Georgian-style transom).

Another form of query-by-example involves the user supplying or selecting an image that is exemplary of the type of content they wish to retrieve. With these approaches, the user is presented with a pseudo-random selection of images of sufficiently diverse content or feature values, and upon selection of one or more images, the system retrieves the set of most closely related images. This process may be repeated to iteratively refine the results until the user is satisfied. Closely related to this approach is query-by-visual-example using relevance feedback. Again, the user is presented with an initial set of pseudo-randomly selected images, for each of which they assign positive or negative feedback as a means of refining their query. This approach is generally less desirable than the others for elevation drawing retrieval, as it often requires considerable user interaction and often performs poorly at interpreting for what image attributes and features the user specified positive and negative feedback.

Query by shape is another common approach to querying.[25,31] In this approach, the user constructs their query image by dragging geometric primitives such as rectangles and circles onto a drawing canvas. In systems where color, texture or spatial layout features are also indexed, multiple shapes may be arranged and filled with colors or model textures that have been extracted from previous query results or from a set of predefined texture samples.[29] The restrictions of using only geometric primitives imposed by this approach limit the user's ability to specify less structured and free-form queries and to apply this approach to more general problem domains.

A less structured approach to querying by shape involves a sketch-based interface with which the user can form queries by drawing free-form objects or using simple geometric shapes. For the retrieval of line drawings, this approach is most suitable as it affords the most familiar interface for users and may include, but is not limited to, predefined shapes, layouts and textures.

4. HIERARCHICAL TEMPORAL MEMORY

In this section, the underlying theory of Hierarchical Temporal Memory (HTM) is introduced. Although HTM may be applied to numerous problem domains where the data consists of both spatial and temporal information (e.g. natural language processing, speech recognition, etc.), the focus of this section is on the processing of visual information.

HTM is a recent paradigm that is based on the memory-prediction framework[9,10] and uses elements of Bayesian networks to model some of the structural and algorithmic properties of the human neocortex. HTM was introduced by Hawkins and Blakeslee,[18] and much of the current HTM research is being conducted by Numenta and researchers at the Redwood Center for Theoretical Neuroscience. Currently, there have been relatively few implementations of HTMs beyond Numenta's initial proof of concept model (NuPIC).[12] An HTM shares many similarities with Bayesian networks, such as the constant sharing of information between nodes, and the use of Belief Propagation, albeit in a modified form. Although HTMs are similar to Bayesian networks, they differ in that HTMs have a clear parent/child relationship, are self-training and can more easily handle time-varying data.[11]

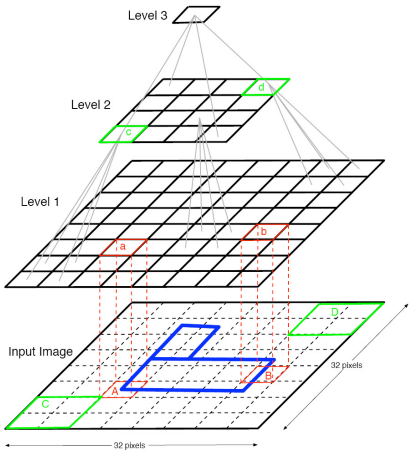

Figure 1. An HTM with 3 levels (adopted from[13]).

The structure of an HTM reflects the nested hierarchical structures found in the world, following the notion that these hierarchies exist in both spatial and temporal dimensions. An HTM is represented as a tree-shaped multi-level hierarchy of nodes, where information can flow in both vertical directions. The exact network configuration (e.g. the number of levels in the hierarchy and the number of nodes at each level) does not affect the underlying theory of HTMs, provided that a clear parent/child relationship is defined between nodes. Various interlayer connection arrangements are permissible (e.g. a child with multiple parents or connections that skip levels of the hierarchy), although certain configurations may provide superior performance for a given problem.

There are two basic functions that an HTM network performs: discovering causes and inferring the causes of novel input. In the HTM literature, a "cause" refers to a persistent and repeating structure in the world, physical or otherwise (e.g. buildings, cats, words, songs, etc.)[19] The first function, discovering causes, involves examining the input data in order to identify patterns that recur in both spatial and temporal dimensions. The second function involves classifying objects in the input data as belonging to a previously discovered cause and, when novel input is encountered, attempting to determine the mostly likely high-level cause that is responsible for the input's occurrence.

The nodes in Level 1 of the HTM receive sensory input directly from one or more sensors (e.g. a CCD in a camera) and use this data to construct a model of the HTM's environment by looking for spatiotemporal correlations in the input data points. Each level 1 node is supplied with sensory data from a small portion of the environment, such that all of the input data is distributed equally across the lowest level nodes without overlap.

Figure 1 illustrates an HTM network with three levels, where the input is a 32x32 pixel image. At level 1, the 32x32 pixel input is received directly from the sensors and distributed across 64 nodes, with each node perceiving a 4x4 pixel area. At level 2, each node receives its input from the outputs of four level 1 nodes, which describe an 8x8 pixel area in the input image.

Whereas the lowest level nodes receive their input from sensory data about the environment, nodes in higher levels of the hierarchy are provided with input comprised of their children's beliefs. Recalling that every node looks for spatial and temporal patterns, the spatial patterns of higher level nodes are formed by the commonly recurring patterns of the beliefs reported by their children, while the temporal patterns are comprised of the recurring changes of their children's beliefs.

Central to HTM theory is that every node in a network shares a common algorithm, regardless of its position in the hierarchy. When new data is submitted to the HTM, each node forms beliefs about the input data by creating two 2-column tables for each of its spatial and temporal beliefs. In each table, the left column enumerates the spatial or temporal patterns that the node has learned, and the right column reports the corresponding probability of the patterns occurring. During each cycle of input data being presented, each node performs two

steps. In the first step, the node assigns to each spatial quantization point, the probability that the current input data matches this point. In the second step, the node searches for common sequences of quantization points, and represents each sequence as a variable. Over time, the node determines the probability that the current input belongs to each each sequence and assigns this probability to each variable. The probability that the input belongs to each sequence variable is added to a vector of probabilities and passed up the hierarchy to serve as input for the node's parent. Each node can also pass belief information down the hierarchy to its children, such as the spatial pattern it anticipates to encounter next, based on the temporal sequence that is believed to be currently occurring. As the hierarchy is ascended, series of input patterns are combined to form a more stable output pattern, while descending the hierarchy transforms a stable pattern into sequences of less stable spatial patterns.

Although each node performs the same basic functions, the hierarchical organization of the network causes the different levels to operate on different phenomenon. The nodes in lower levels deal with simple events that change quickly and occupy smaller spatial areas, while nodes in higher levels deal with multiple events that were sensed by lower level nodes, and thus are influenced by a greater range of data. Specifically, higher nodes sense more complex causes, namely patterns of patterns, which evolve less quickly and exist in larger spatial areas. This property is similar to the basic principles of Slow Feature Analysis, which have shown that invariant features can be learned using a hierarchy that uses temporal "slowness" as a fundamental principle for learning.[42]

HTMs are generative in nature, in that they have the ability to generate data that can be used to make predictions about what will occur in the future.[11] This happens as part of the learning algorithm used by the nodes, which store sequences of patterns that are likely to occur. Predicting what is likely to happen at the next time step is performed by combining the node's known sequences of patterns with the current input and checking the known sequences to determine what is likely to happen next.

A common way for conventional recognition systems to classify visual objects is to use template matching, wherein the system stores prototypical and transformed representations of each known object, and classifies each candidate object by comparing it against the templates. This approach is impractical for large scale problems, because the memory requirements grow exponentially as the problem space increases. Further, template matching approaches also perform poorly when noisy and novel objects are encountered. HTMs however, are much more scalable, as a vision system built with an HTM does not perform transformations on the candidate objects to match it to a template, and does not store prototypes in this manner. Rather, each node in an HTM network stores a fixed number of quantization points, where each quantization point represents a commonly seen pattern. Typically, each node maintains 50–100 quantization points, and although this number does not change during training, the pattern that each point represents can be modified.

During each cycle, a node determines the distance between the input data and each quantization point. Lower level nodes do not know anything about higher level causes since they can only perceive a small area of the input. The causes discovered by lower level nodes are causes of low complexity, such as edges, lines and corners, which can serve as components in higher level causes. This allows for scalable and efficient memory usage, as the memory used to store low level causes is shared by the high level causes. However, a shortcoming of this paradigm is that the network has difficulty learning to recognize new objects that are not composed of previously learned sub-objects.

HTMs can be trained using either supervised or unsupervised learning, although even in supervised scenarios, all of the nodes in the network except those at the top level learn in an unsupervised manner. In unsupervised learning, the temporal arrangement of the training data acts as a teacher to show which spatial beliefs belong together. The option for supervised training of the nodes in the top level involves associating each quantization point in the top level with a category index that provides more meaningful labels than the network internally provides (i.e. an index number) to each quantization point.

5. CBIR USING HTM

In this section, the HTMCBIR system is introduced that uses Numenta's NuPic platform[13] as a basis to perform indexing, querying and retrieval of line drawings. The system is based on the NuPic PicturesDemo, which creates an HTM network trained to recognize 48 categories of binary line drawings (Figure 2). The system was

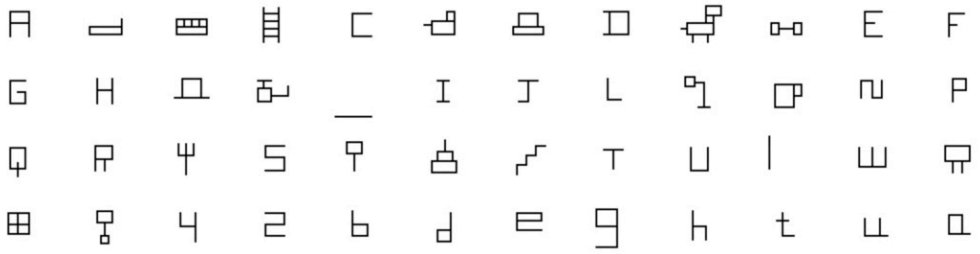

Figure 2. Iconic examples of the 48 image categories that the network was trained to recognize.

implemented and subjected to experimentation on a Dell Latitude D810 with a 2.13 GHz processor and 1 GB of RAM under Windows XP.

The network topology of the HTMCBIR system consists of four levels, where the nodes in the lowest level operate directly on the pixel data, and nodes in each subsequent higher level operate on the belief vectors produced by the previous level. Once the network topology is established, the HTM is trained by presenting it with images of different object categories. Each training image is a 32x32 pixel binary bitmap image belonging to the set of 945 training images that are divided into 48 categories (see Table 1). Training proceeds in a supervised manner, with each training image of a given category being stored in a directory of that category's name (e.g. /cat/cat3.bmp), which is used by only the top level nodes. Once training is complete, the system is ready to receive and process queries. For simplicity, the indexed images are stored in their native directory structure rather than in a more complex database system, although such an improvement may be easily implemented and is considered as future work.

Category	Images	Category	Images	Category	Images	Category	Images
a	27	h	24	q	19	spoon	13
bed	37	hat	15	r	18	stack	30
bus	11	helicopter	18	rake	15	steps	20
c	18	horz_line	15	s	18	t	14
cat	26	i	31	small_a	9	u	17
computer	19	j	19	small_b	17	vert_line	27
d	17	l	13	small_d	13	w	22
dog	35	ladder	14	small_e	19	whiteboard	20
dumbbell	25	lamp	20	small_g	14	window	24
e	33	mug	31	small_h	19	wineglass	19
f	22	n	11	small_t	16	y	13
g	20	p	22	small_u	10	z	16

Table 1. Number of training images per object category for HTMCBIR.

Two methods of specifying a query are provided in HTMCBIR. The first falls under the iconic indexes paradigm, in which the user selects an iconic representation of an object category from the "Training Images" panel of the main window (Figure 3). The second method allows the user to use their mouse to sketch a line drawing query on the 32x32 drawing canvas. With both methods, once the query is entered into the drawing canvas, the user clicks the "Recognize Picture" button and the query is submitted to the system for recognition and classification. These two methods for submitting a query offer several advantages over other approaches. First, the majority of elevation drawings are grey scale or binary images, and thus color- and texture-based approaches are largely not applicable. These methods also allow the user to visually specify their query in terms of object shape and spatial layout, which provides a more usable and intuitive interface that is not influenced by the user's language or vocabulary.

To retrieve images based on the query image, HTMCBIR attempts to infer the category of the sketch based

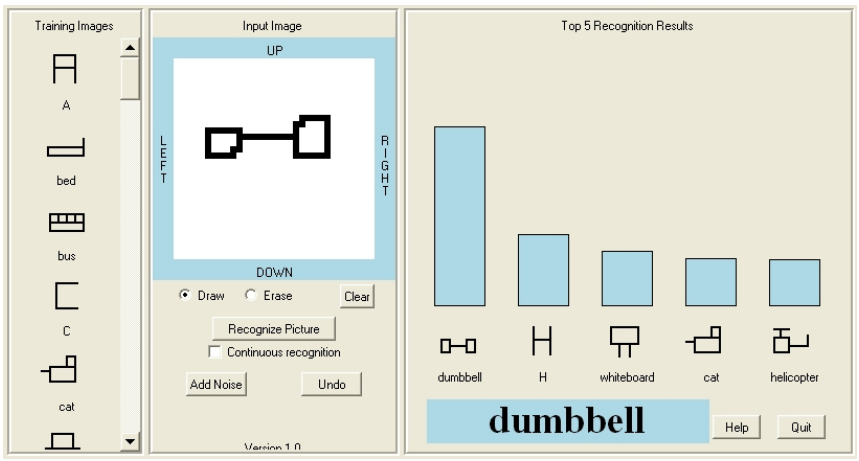

Figure 3. Main window of the HTMCBIR application, showing the sketch-based interface where the user has drawn a dumbbell, and the recognition results for the query image. Iconic indexes can also be selected from the "training images" frame.

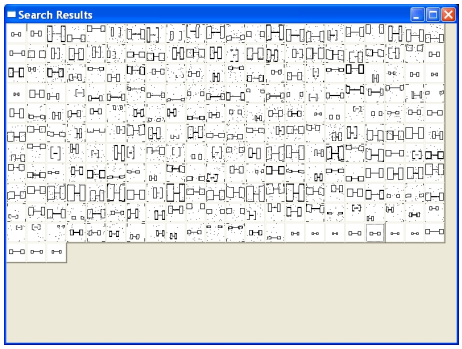

Figure 4. The retrieved images from the "dumbbell" category.

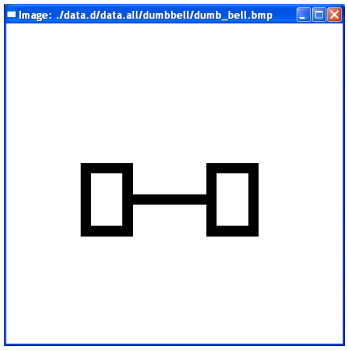

Figure 5. A retrieved dumbbell image and its location on disk.

from the set of learned categories. The results of the recognition process are displayed in the main window as the five best matching object categories, with each possible category's degree of certainty being indicated by a bar chart (Figure 3). If the correct category is not included in this list, the user alters their query and has the system attempt to recognize it again. Once the intended category is included in the matching results, image retrieval proceeds by the user clicking on the category's icon or label under the bar chart. This opens a secondary window containing thumbnail versions of each stored image in that category (Figure 4). Clicking on one of the thumbnails displays a zoomed-in version of the selected image and its location on disk (Figure 5).

6. EXPERIMENTAL RESULTS

To measure the recognition accuracy of HTMCBIR for visual queries, the system was evaluated on three criteria using three experiments: recognition accuracy for a large hand-drawn corpus; accuracy with noisy data; and accuracy with rotated data. The first two experiments use the same HTM network that is trained using 953 clean 32x32 pixel images from 48 categories, while the third experiment uses the same training set, but applies to it varying degrees of rotation. The following subsections introduce and discuss the results of each of the three experiments. For the remainder of this article, "recognition accuracy" is defined as the number of correctly classified query images divided by the total number of testing images.

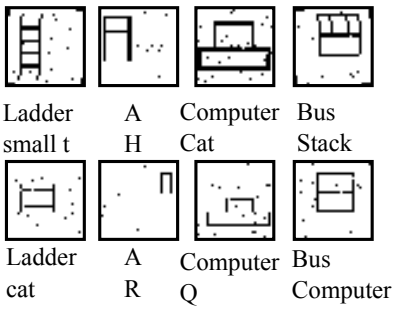

Figure 6. Examples of errors from the query recognition test set. The label in the first row below each image indicates its true class, and the second row is the incorrect label that was assigned.

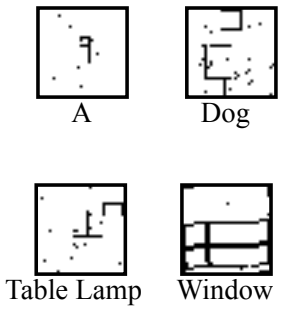

Figure 7. Examples of difficult, but correctly classified query images from the test set.

6.1. Recognition Accuracy with a Testing Corpus

The first experiment evaluates the HTMCBIR's ability to recognize visual queries using a test set of 8941 distorted images and aims to measure the generalization ability of the system for a large corpus of testing data. The hand-drawn test images are based on the training images, but have been distorted with warping, additive noise, occlusions, and affine transformations relative to the training set. A single run of this experiment was conducted, as static data sets are used for training and testing.

Using 8941 images in the distorted data set, a recognition accuracy of 69.7% was observed, wherein 6232 of the query images were correctly recognized. Figure 6 shows several query images that were incorrectly recognized. Below each image, the label in the first row indicates the true class of the query image, and the label in the second row indicates the label assigned by the system. Comparing these images with the iconic indexes shown in Figure 2, queries in the first row are more likely to be correctly recognized by the reader, and were misclassified due to the presence of noise and their variation from the training images. Queries in the second row of images in Figure 6 however, are likely to be also misclassified by human viewers, as they bare little resemblance to the other images in their respective categories, and thus present a very difficult problem for the system. Figure 7 provides examples of test image queries that were correctly classified by the system, despite their low resemblance to the training images from that category. Several of these images are missing large sections of the contained object, and their correct classification reflects the system's ability to recognize noisy, distorted and incomplete queries.

6.2. Recognition Accuracy with Additional Noise

The second experiment evaluates the influence that varying the amount of noise in the testing data has on the system's ability to recognize a query image. For this experiment, two types of noise are added to the testing images: thermal noise and spatial noise. The `thermalNoise` parameter specifies the probability that the value of any given pixel in the image will be inverted (i.e. changed from 0 to 1 or vice versa). Spatial noise involves scanning through an image looking for black pixels, which when found, are moved in a random direction with a distance determined by the parameter `spatialNoiseStddev`, with probability `spatialNoise`. For this experiment, `spatialNoiseStddev` is set to 1.0, and `thermalNoise` and `spatialNoise` are varied.

As this experiment is deterministic and employs the same training and testing data during each trial, only a single run was conducted. Figure 8 summarizes the results of this experiment, where the amount of spatial and thermal noise was varied between 0% and 40%. With no spatial and thermal noise, the recognition accuracy of the network is 99.3%. The influence of spatial noise is less pronounced than is thermal noise, as the former causes a set pixel to be translated a small distance, while the latter involves inverting random pixels. That is, with `thermalNoise`, P_t, each pixel is inverted with probability P_t, which for larger values causes a significant change in the image appearance, whereas `spatialNoise`, P_s only causes moderate distortion of the object itself and has less influence on the overall image. With $P_s = 0.4$ and $P_t = 0.0$, the system is shown to be robust to spatial noise, achieving a recognition accuracy of 69.32%, whereas with $P_s = 0.0$ and $P_t = 0.4$, the accuracy

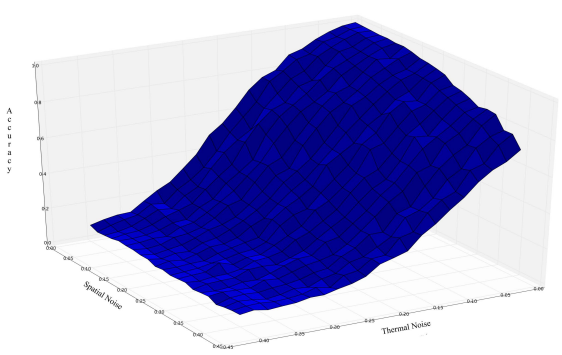

Figure 8. Influence of varying the probability of thermal and spatial noise on query recognition accuracy.

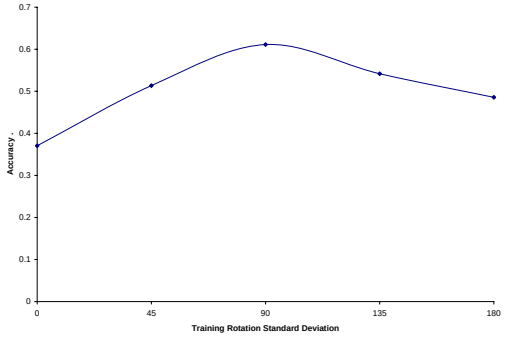

Figure 9. Influence that rotating the training images has on recognition accuracy of rotated query images.

is only 7.51%. The system's robustness to spatial noise is particularly valuable for CBIR applications, as the distortion caused by spatial noise approximates the variability found in sketch-based queries.

6.3. Recognition Accuracy with Rotation

The third experiment evaluates the system's ability to learn rotation invariance for recognizing visual queries. Unlike the previous two experiments, the HTM network used here is trained on images that have been subjected to varying degrees of rotation and then tested against rotated images. This experiment generates different data for each trial, and thus a total of three runs were conducted, the average results of which are given in Figure 9. Each training image was rotated an amount determined by the parameter `trainingRotationStddev`, σ_r. Each training image was shown to the network a total of four times, with each image being rotated a random amount determined from a distribution where $\mu = 0$ and $\sigma = \sigma_r$. As would be expected, the training time required by presenting each image to the system four times was approximately four times longer than for the other two experiments.

The highest recognition accuracy was achieved with $\sigma_r = 90°$, where the system's mean recognition accuracy was 61.06% with a standard deviation of 7.78E-05 across three runs. The chart of the accuracy produced by the remaining values of σ_r (Figure 9) approaches a normal distribution with increasing runs, with values of σ_r greater or less than 90° producing lower accuracies. This result is somewhat surprising, as it was initially expected that $\sigma_r = 180°$ would produce the highest accuracy, as this value exposes the system to each object in the widest range of orientations during training. One hypothesis to explain this phenomenon is that the system requires more than four exposures to each training image to be able to recognize images with $\sigma_r > 90°$. This hypothesis could be tested by repeating the experiment, but with a network trained using more rotations of each image. A second hypothesis is that, due to the small scale of each image, rotating the object produces an image with considerable distortion, particularly when σ_r is not a multiple of 90°. This occurs because the straight lines present in the original image become "zigzag" lines when rotated, for example, 45°. This hypothesis could be tested by repeating the experiment with larger images (e.g. 64x64 or 128x128 pixels) and determining if the phenomenon is again observed.

7. CONCLUSION

The CBIR system introduced here extends Numenta's NuPic implementation of HTM for the indexing and retrieval of line drawings. Due to time and computer hardware constraints, the system was implemented using 32x32 pixel binary images in place of large scale elevation drawings. To evaluate its ability to index these line drawing images and to recognize visual queries, three experiments were conducted. The first experiment measured the system's recognition ability using a corpus of 8941 test images that were distorted with noise, warping, occlusions, and affine transformations, finding that the HTMCBIR system achieved a recognition accuracy of 69.7%.

The second experiment evaluated the system's ability to recognize noisy testing data. The system was shown to be robust to spatial noise, achieving a recognition accuracy of 87.86% with a 24% probability of spatial noise, and 69.32% with a 40% probability of spatial noise. The third experiment evaluated the system's ability to learn rotational invariance of objects by training and testing a network using rotated images. Within a training rotational standard deviation of 90°, the system achieved a mean accuracy of 61.06% and standard deviation of 7.78E-05 across three runs.

With additional training data and further experimentation to determine the optimal network parameter values, it is believed that the HTMCBIR system is capable of achieving a higher recognition accuracy. The purpose of this implementation is a proof of concept to show that HTM is flexible enough to provide efficient and accurate indexing of line drawings. There are numerous directions for future work in this area, which are discussed in the following section.

8. FUTURE WORK

There are several directions that future work may take on this approach to CBIR. First, the system currently operates on binary images rather than grey scale or color images. This decision was made because the majority of elevation drawings are created with black ink on white paper, and thus most digitized drawings are stored as binary images. For grey scale digitized elevation drawings, several binarization algorithms have been shown to accurately convert grey scale elevation drawings to a binary format with minimal loss of information.[3] For these images, the substantial reduction of computational complexity resulting from the use of binary images far outweighs the value of operating on grey scale images. Note however, that the current version of NuPic is capable of performing classification on grey scale photographs, and this system could be easily modified for other image formats.

The second direction for future work pertains to the data set itself. Typically, the resolution of digital elevation drawings is much larger than the 32x32 pixel images used here. Additionally, elevation drawing archives often contain large numbers of high resolution images (e.g. the United States Library of Congress's Historic American Building Survey has approximately 30,000 such images[1]). The smaller data set employed for these experiments was selected due to the substantial time and resources required for training a system on large data sets with high resolution images. Future work may investigate the scalability of HTM and evaluate its suitability for processing large data sets.

Another direction for future work is to index images containing more than a single object. This is an open problem in the HTM literature, as there has not yet been an implementation that can incorporate our biological understanding of human attentional mechanisms. The attention problem for multiple objects is the subject of this author's Ph. D. thesis work, and will be examined in forthcoming research.

9. SUMMARY

This article has surveyed several prominent indexing features and querying interfaces for CBIR in the context of architectural elevation drawing databases. For indexing, these approaches often fail to capture the semantic information contained in the images by using image features and attributes that greatly differ from those used by the human vision system for object recognition and clustering of related images. Hierarchical Temporal Memory is a biomimetic approach to the vision problem, that in the context of CBIR, aims to "bridge the semantic gap" by translating the low-level features to high-level concepts that are more easily understood by the user and allow them to specify queries using their own terminology. The HTMCBIR system presented here, which extends Numenta's NuPic implementation, has been shown to provide promising results for indexing and recognizing small binary images. The querying interface of HTMCBIR allows the user to quickly and easily specify a query, and the query image recognition algorithm was shown to be robust to spatial noise, occlusions, distortion, and affine transformations.

REFERENCES

1. Historic American Buildings Survey/Historic American Engineering Record (HABS/HAER). http://memory.loc.gov/ammem/collections/habs_haer, 2007.
2. Ilaria Bartolini and Marco Patella. WARP: Accurate retrieval of shapes using phase of fourier descriptors and time warping distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(1):142–147, 2005. Member-Paolo Ciaccia.
3. Bruce Bobier. Evaluation of binarization algorithms for preprocessing elevation drawings. *Unpublished*, University of Guelph, Guelph, ON, March 2007.
4. Vittorio Castelli and Lawrence D. Bergman. *Image Databases: Search and Retrieval of Digital Imagery*. Wiley-Interscience, 1st edition, December 2001.
5. S. L. Feng, R. Manmatha, and V. Lavrenko. Multiple Bernoulli relevance models for image and video annotation. volume 2, pages II–1002–II–1009 Vol.2, 2004.
6. Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley, Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petkovic, David Steele, and Peter Yanker. Query by image and video content: the QBIC system. pages 7–22, 1997.
7. Manuel J. Fonseca. *Sketch-Based Retrieval in Large Sets of Drawings*. PhD thesis, Instituto Superior Tecnico, July 2004.
8. Charles Frankel, Michael J Swain, and Vassilis Athitsos. WebSeer: An image search engine for the world wide web. Technical report, University of Chicago, Chicago, IL, USA, 1996.
9. Saulius Garalevicius. Memory-prediction framework for pattern recognition: Performance and suitability of the Bayesian model of visual cortex. Paper accepted for FLAIRS-20, 2007.
10. Dileep George and Jeff Hawkins. Invariant pattern recognition using Bayesian inference on hierarchical sequences. Technical report, Redwood Neuroscience Institute, 2005.
11. Dileep George and Bobby Jaros. Hierarchical Temporal Memory – comparison with existing models (ver. 1.01). Numenta Inc. Whitepaper, 2007.
12. Dileep George and Bobby Jaros. Hierarchical Temporal Memory – Zeta1 algorithms reference. Numenta Inc. Whitepaper, 2007.
13. Dileep George and Bobby Jaros. The HTM learning algorithm. Numenta Inc. Whitepaper, 2007.
14. Theo Gevers and Arnold W. M. Smeulders. Color based object recognition. In *ICIAP '97: Proceedings of the 9th International Conference on Image Analysis and Processing-Volume I*, pages 319–326, London, UK, 1997. Springer-Verlag.
15. Ilias Grinias, Nikos Komodakis, and Georgios Tziritas. Bayesian region growing and MRF-based minimization for texture and colour segmentation. In *Image Analysis for Multimedia Interactive Services, 2007. WIAMIS '07.*, page 20, 2007.
16. Mark D. Gross and Ellen Yi-Luen Do. Diagram query and image retrieval in design. In *ICIP '95: Proceedings of the 1995 International Conference on Image Processing*, volume 2, pages 2308–2313, Washington, DC, USA, 1995. IEEE Computer Society.
17. James Hafner, Harpreet S. Sawhney, Will Equitz, Myron Flickner, and Wayne Niblack. Efficient color histogram indexing for quadratic form distance functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(7):729–736, 1995.
18. Jeff Hawkins and Sandra Blakeslee. *On Intelligence*. Times Books, 2004.
19. Jeff Hawkins and Dileep George. Hierarchical Temporal Memory – concepts, theory, and terminology. Numenta Inc. Whitepaper, 2007.
20. Ming-Kuei Hu. Visual pattern recognition by moment invariants. *Information Theory, IEEE Transactions on*, 8(2):179–187, 1962.
21. J. Huang, S. Kumar, M. Mitra, W. Zhu, and R. Zabih. Image indexing using color correlograms. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 762–768, 1997.
22. H Kauppinen, T Seppanen, and M Pietikainen. An experimental comparison of autoregressive Fourier-based descriptors in 2D shape classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):201–207, February 1995.

23. B.B. Kimia. Shape representation for image retrieval. In V. Castelli and L.D. Bergman, editors, *Image Databases; Search and Retrieval of Digital Imagery*, chapter 13, pages 345–372. Wiley, New York, 2002.
24. Michael S. Lew, Nicu Sebe, Chabane Djeraba, and Ramesh Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Transactions on Multimedia Computer Communication Applications*, 2(1):1–19, February 2006.
25. Aiyesha Ma and Ishwar K. Sethi. Local shape association based retrieval of infrared satellite images. In *ISM '05: Proceedings of the Seventh IEEE International Symposium on Multimedia*, pages 551–557, Washington, DC, USA, 2005. IEEE Computer Society.
26. Babu M. Mehtre, Mohan S. Kankanhalli, and Wing F. Lee. Shape measures for content based image retrieval: A comparison. *Information Processing & Management*, 33(3):319–337, May 1997.
27. P.A. Mlsna and N.M. Sirakov. Intelligent shape feature extraction and indexing for efficient content-based medical image retrieval. In *IEEE Southwest Symposium on Image Analysis and Interpretation*, pages 172–176, 2004.
28. Surya Nepal and M. V. Ramakrishna. Query processing issues in image(multimedia) databases. In *ICDE '99: Proceedings of the 15th International Conference on Data Engineering*, pages 22–29, Washington, DC, USA, 1999. IEEE Computer Society.
29. P. Pala and S. Santini. Image retrieval by shape and texture. *Pattern Recognition*, 32:517–527, 1999.
30. T. Pavlidis. Polygonal approximations by Newton's method. *IEEE Transactions on Computing*, 26(8):800–807, 1977.
31. Ediz Saykol, Ugur Gudukbay, and Ozgur Ulusoy. A histogram-based approach for object-based query-by-shape-and-color in image and video databases. *Image and Vision Computing*, 23:1170–1180, November 2005.
32. Lambert Schomaker, Edward de Leau, and Louis Vuurpijl. Using pen-based outlines for object-based annotation and image-based queries. In *VISUAL '99: Proceedings of the Third International Conference on Visual Information and Information Systems*, pages 585–592, London, UK, 1999. Springer-Verlag.
33. Nicu Sebe and Michael S. Lew. Color-based retrieval. *Pattern Recognition Letters*, 22(2):223–230, 2001.
34. John R. Smith and Shih-Fu Chang. Visually searching the web for content. *IEEE MultiMedia*, 4(3):12–20, 1997.
35. Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.
36. S.L. Tanimoto. *An Iconic/Symbolic Data Structuring Scheme*. Academic Press, New York, 1976.
37. N. Vasconcelos. Minimum probability of error image retrieval. *IEEE Transactions on Signal Processing*, 52(8):2322–2336, 2004.
38. R. Veltkamp and M. Hagedoorn. State-of-the-art in shape matching. Technical Report UU-CS-1999-27, Utrecht University, the Netherlands, 1999.
39. R. Veltkamp and M. Tanase. Content-based image retrieval systems: A survey. Technical report, Utrecht University, the Netherlands, 2002.
40. C. Vertan and N. Boujemaa. Embedding fuzzy logic in content based image retrieval. In *Fuzzy Information Processing Society, 2000. NAFIPS. 19th International Conference of the North American*, pages 85–89, 2000.
41. Jules Vleugels and Remco C. Veltkamp. Efficient image retrieval through vantage objects. In *VISUAL '99: Proceedings of the Third International Conference on Visual Information and Information Systems*, pages 575–584, London, UK, 1999. Springer-Verlag.
42. Laurenz Wiskott and Terrence Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002.
43. Shao Ying Zhu and Gerald Schaefer. Thermal medical image retrieval by moment invariants. In *5th International Symposium on Biological and Medical Data Analysis*, pages 182–187, 2004.