Preprint for Intelligent Decision Technologies 10(2) (February 2016):165-181. DOI: 10.3233/IDT-150246.

Measuring the Nearness of Layered Flow Graphs: Application to Content Based Image Retrieval

Kanwarpreet Kaur[1], Sheela Ramanna[1], and Christopher Henry[1]

[1]Department of Applied Computer Science, University of Winnipeg, Manitoba R3B 2E9 Canada

kan.preet64@gmail.com, s.ramanna@uwinnipeg.ca, ch.henry@uwinnipeg.ca

Abstract

Rough set based flow graphs represent the flow of information for a given data set where branches of these could be constructed as decision rules. However, in the recent years, the concept of flow graphs has been applied to perceptual systems (also called perceptual flow graphs) where they play a vital role in determining the nearness among disjoint sets of perceptual objects. Perceptual flow graphs were first introduced to represent and reason about sufficiently near visual points in images. In this paper, we have given a practical implementation of flow graphs induced by a perceptual system, defined with respect to digital images, to perform Content-Based Image Retrieval(CBIR). Results are generated using the SIMPLicity dataset, and our results are compared with the near-set based tolerance nearness measure(tNM).

Keywords: Content Based Image Retrieval, granular computing, flow graphs, near sets, perceptual system, rough sets.

1 Introduction

Rough set based flow graphs first introduced by Pawlak in [24, 3] to model information flow for a given data set with branches representing decision rules. The branches of flow graphs can be constructed as a decision rule and have three coefficients: strength, certainty and coverage associated with each branch. The entire flow graph can be viewed as a learning structure. Recent work on rough set based flow graphs includes theoretical and algorithmic aspects of Pawlak flow graphs [6, 27, 30, 29, 31, 1, 41] as well as practical applications such as music retrieval [14, 5], survival analysis [21], association rule [2], data mining [42] and granular computing [15].

The basic structure used in rough set-based flow graphs is an information system. However, in this work, we use a near set flow graph with a perceptual system based on near sets [33, 37, 44, 9] as its basic structure. The formal model for the near set based flow graphs was introduced in [35] and elaborated in [40].

A perceptual system is a specialized form of information system consisting of a set of objects equipped with a family of probe functions. These probe functions give rise to a number of perceptual relations between objects of a perceptual system [45]. This approach is useful when decisions on nearness are made in the context of a perceptual system, i.e., a system consisting of objects and our perceptions of what constitutes features that best describe these objects. This is especially important in image retrieval [12].

In a perceptual flow graph (PFG) induced by a perceptual system, a node in the graph is an object in a perceptual system with normalized flows derived from probe functions. A distinctive feature of graphs induced by perceptual systems are layers. A layer consists of nodes belonging to a single feature. Hence, layers are partitions induced by probe function values where nodes within a layer are not connected. An important characteristic of such flow graphs is that layers greatly influence nearness measure. Different ordering among set of probe functions generate different results.

Perceptual flow graphs were first introduced in [35] to represent and reason about sufficiently near visual points in images. In [40], a framework for extended layered perceptual flow graphs was established, where analysis of such graphs was performed using near set theory. In [36], this framework was extended to include set of points between pairs of digital image flow graphs. In this paper we introduce, i) a perceptual flow graph algorithm (PFG) to determine nearness between two disjoint perceptual systems, ii) a binning method to discretize the real-valued domain of feature values, iii) an efficient implementation of the algorithm by determining the best combination of a reduced set of features using entropy-based gain ratio measure iv) experimental comparison with the tolerance nearness measure results reported in [11] (see also [10]).

The choice of a perceptual system for CBIR application was especially important for two main reasons: i) probe functions are defined in terms of features of pixels in digital images (e.g., colour, texture, edges, and moments, ii) retrieval task can be considered as measuring nearness between two disjoint perceptual systems (i.e., two digital images). In this paper, the term *nearness* is used in the context of PFGs whereas the term *similarity* is used in the context of a specific application such as CBIR.

The paper is organized as follows. In Sect. 2, we give formal definitions for near sets and rough set based flow graphs. In Sect. 3, we discuss research related to this paper. In Sect. 4, we present formal definitions for perceptual flow graphs and an application to CBIR. A discussion of the results is given in Sect. 6.

2 Preliminaries

Underlying perceptual flow graphs, is the notion of an ordered perceptual system. In this section, we give basic definitions of near set theory, rough set based flow graphs and extended layered flow graphs.

2.1 Perceptual System

The basic structure which underlies near set theory is a perceptual system [38].

Definition 1 ([9]) Perceptual Object. *A* perceptual object *is anything that has its origin in the physical world i.e. it possesses some characteristics that are observable to the senses.*

Definition 2 ([32]) Feature. *A* feature *characterizes some aspect of the makeup of a perceptual object.*

Definition 3 ([33, 34]) Probe Function. *A* probe function *is the real-valued function that represents the features of a perceptual object.*

In this work, probe functions are defined in terms of digital images such as: colour, texture, gradient and spatial orientation. In relation to the near set theory, probe functions play an important role by determining the similarity and dissimilarity among given set of images, thereby finding that, if two objects are associated with same pattern or not.

Definition 4 ([9]) Perceptual System. *A* perceptual system *is a pair $\langle O, \mathbb{F} \rangle$, where O is a nonempty set of perceptual objects and F is a countable set of probe functions $\phi_i : O \to \mathbb{R}$.*

Definition 5 ([10]) Object Description. *Let $\langle O, \mathbb{F} \rangle$ be a perceptual system and $\mathcal{B} \subseteq \mathbb{F}$ be a set of probe functions. Then, the* description *of the perceptual object $x \in O$ is given in terms of the feature vector :*

$$\phi_{\mathcal{B}}(x) = \phi_1(x), \phi_2(x), ..., \phi_i(x), ..., \phi_l(x),$$

where l is the length of the feature vector $\phi_{\mathcal{B}}(x)$ and $\phi_i(x)$ in $\phi_{\mathcal{B}}(x)$ is a probe function value that is a part of the perceptual object $x \in O$.

Definition 6 ([22, 38]) Perceptual Indiscernibility Relation. *Let $\langle O, \mathbb{F} \rangle$ be a perceptual system. For every $\mathcal{B} \subseteq \mathbb{F}$, the* perceptual indiscernibility relation $\sim_{\mathcal{B}}$ *is defined as:*

$$\sim_{\mathcal{B}} = \{(x, y) \in O \times O : \forall \phi_i \in \mathcal{B}.\ \phi_i(x) = \phi_i(y)\}.$$

Definition 7 ([10]) Equivalence Class. *Let $\langle O, \mathbb{F} \rangle$ be a perceptual system and let $x \in O$. For a set $\mathcal{B} \subseteq \mathbb{F}$, an* equivalence class *is defined as:*

$$x_{/\sim_{\mathcal{B}}} = \{x' \in O | x' \sim_{\mathcal{B}} x\}.$$

Definition 8 ([38]) Nearness Relation. *Let $\langle O, \mathbb{F} \rangle$ be a perceptual system and let $X, Y \subseteq O$. A set X is near to set Y within the perceptual system $\langle O, \mathbb{F} \rangle$ $(X \bowtie_{\mathbb{F}} Y)$ if, and only if, there are $\mathcal{B}_1$, $\mathcal{B}_2 \subseteq \mathbb{F}$ and $\phi_i \in \mathbb{F}$ and there are $A \in O_{/\sim \mathcal{B}_1}$, $B \in O_{/\sim \mathcal{B}_2}$, $C \in O_{/\sim \phi_i}$ such that $A \subseteq X, B \subseteq Y$, and $A, B \subseteq C$. If a perceptual system is understood, than a set X is near to set Y.*

Definition 9 ([38]) Perceptual Near Sets. *Let $\langle O, \mathbb{F} \rangle$ be a perceptual system and let $X, Y \subseteq O$ denote disjoint sets. Sets X, Y are near sets if, and only if, $X \bowtie_{\mathbb{F}} Y$.*

2.2 Rough Set Based Flow Graphs

In this section, we introduce formal notation for flow graphs based on rough sets.

Definition 10 ([24]) *Let $G = (N, B, \varphi)$ be a directed, acyclic, finite graph, where N is a set of nodes, $B \subseteq N \times N$ a set of* directed branches, *$\varphi : B \to \mathbb{R}^+$ a* flow function *and $\mathbb{R}^+$ denotes a set of non-negative real numbers.*

If $(x, y) \in B$ then x is an *input* of node y denoted by $I(y)$ and y is an *output* of node x denoted by $O(x)$. Next, *input* and *output* of a flow graph G are defined respectively by $I(G) = \{x \in N : I(x) = \emptyset\}$ and $O(G) = \{x \in N : O(x) = \emptyset\}$.

These inputs and outputs of G are called *external nodes* of G whereas other nodes are called *internal nodes* of G. If $(x, y) \in B$ then we call (x, y) a *throughflow* from x to y. We will assume in what follows that $\varphi(x, y) \neq 0$ for every $(x, y) \in B$. With every node x of a flow graph G, we have its associated *inflow* and *outflow* respectively as: $\varphi_+(x) = \sum_{y \in I(x)} \varphi(y, x)$ and $\varphi_-(x) = \sum_{y \in O(x)} \varphi(x, y)$. Similarly, an *inflow* and an *outflow* for the flow graph G are defined as: $\varphi_+(G) = \sum_{x \in I(G)} \varphi_-(x)$ and $\varphi_-(G) = \sum_{x \in O(G)} \varphi_+(x)$. We assume that for any internal node x, $\varphi_-(x) = \varphi_+(x) = \varphi(x)$, where $\varphi(x)$ is a *throughflow* of node x. Similarly then, $\varphi_-(G) = \varphi_+(G) = \varphi(G)$ is a *throughflow* of graph G.

2.3 Normalized Flow Graphs

Definition 11 ([24]) *Let $\mathcal{G} = (N, B, \varphi, \sigma)$ be a normalized flow graph, where N is a set of nodes, $B \subseteq N \times N$ a set of* directed branches, *$\varphi : B \to \mathbb{R}^+$ and σ: $B \to [0, 1]$ a* normalized flow *between nodes.*

With every node x of a normalized flow graph $\mathcal{G}$, the associated *normalized inflows* and *outflows* are defined as: $\sigma_+(x) = \frac{\varphi_+(x)}{\varphi(G)} = \sum_{y \in I(x)} \sigma(y,x)$ and $\sigma_-(x) = \frac{\varphi_-(x)}{\varphi(G)} = \sum_{y \in O(x)} \sigma(x,y)$. For any internal node x, it holds that $\sigma_+(x) = \sigma_-(x) = \sigma(x)$, where $\sigma(x)$ is a *normalized throughflow* of x. Similarly, *normalized inflows* and *outflows* for the flow graph $\mathcal{G}$ are defined as: $\sigma_+(G) = \frac{\varphi_+(G)}{\varphi(G)} = \sum_{x \in I(\mathcal{G})} \sigma_-(x)$ and $\sigma_-(\mathcal{G}) = \frac{\varphi_-(G)}{\varphi(G)} = \sum_{x \in O(G)} \sigma_+(x)$. It also holds that $\sigma_+(\mathcal{G}) = \sigma_-(\mathcal{G}) = \sigma(\mathcal{G}) = 1$.

For any branch (x,y) the *strength* σ is defined as:

$$\sigma(x,y) = \frac{\varphi(x,y)}{\varphi(G)},$$

where $0 \leq \sigma(x,y) \leq 1$. With every branch (x,y) of a normalized flow graph $\mathcal{G}$, the *certainty* and the *coverage* of (x,y) are defined respectively as:

$$cer(x,y) = \frac{\sigma(x,y)}{\sigma(x)},$$

and

$$cov(x,y) = \frac{\sigma(x,y)}{\sigma(y)},$$

where $\sigma(x), \sigma(y) \neq 0$ and is defined as:

$$\sigma(x) = \frac{\text{Size of equivalence class}}{\text{Inflow}}.$$

In accordance with the previous works [6], here are some consequence properties:

$$\sum_{y \in O(x)} cer(x,y) = 1 \quad \text{and} \quad \sum_{x \in I(y)} cov(x,y) = 1.$$

In addition,

$$\sigma(x) \ = \sum_{y \in O(x)} cer(x,y)\sigma(x) \ = \sum_{y \in O(x)} \sigma(x,y),$$

and

$$\sigma(y) \ = \sum_{x \in I(y)} cov(x,y)\sigma(y) \ = \sum_{x \in I(x)} \sigma(x,y),$$

have a form of total probability theorem, whereas

$$cer(x,y) = \frac{cov(x,y)\sigma(y)}{\sigma(x)},$$

and

$$cov(x,y) = \frac{cer(x,y)\sigma(x)}{\sigma(y)},$$

are Bayes' rules.

A (directed) *path* from x to y ($x \neq y$), denoted by $[x \dots y]$, is a sequence of nodes $x_1, \dots, x_n$ such that $x_1 = x$ and $x_n = y$ and $(x_i, x_{i+1}) \in B$ for every i, $1 \leq i \leq n-1$. The certainty of the path $[x_1 ... x_n]$ is defined as

$$cer[x_1..x_n] = \prod_{i=1}^{n-1} cer\left(x_i, x_{i+1}\right), \tag{1}$$

the coverage of the path $[x_1 ... x_n]$ is defined as

$$cov[x_1 ... x_n] = \prod_{i=1}^{n-1} cov\left(x_i, x_{i+1}\right),$$

and strength of the path $[x_1 ... x_n]$ is defined as

$$\sigma[x_1 ... x_n] = \sigma(x_1) cer[x_1 ... x_n] = \sigma(x_n) cov[x_1 ... x_n]. \tag{2}$$

If $[x \dots y]$ is a path such that x and y are input and output of the graph $\mathcal{G}$, respectively, then $[x \dots y]$ will be referred to as a *complete path.*

Since normalized flow graphs are composed of internal and external nodes (inputs and outputs), if we only focus on input and output, then we require the concept of a complete connection. The set of all complete paths from x to y ($x \neq y$) in $\mathcal{G}$, denoted by $\langle x, y \rangle$, is a *complete connection* of $\mathcal{G}$ determined by nodes x and y. For every complete connection $\langle x, y \rangle$, its associated *certainty, coverage* and *strength of the complete connection* $\langle x, y \rangle$ are:

$$cer\left\langle x, y \right\rangle = \sum_{[x \dots y] \in \langle x, y \rangle} cer[x \dots y],$$

$$cov\left\langle x, y \right\rangle = \sum_{[x \dots y] \in \langle x, y \rangle} cov[x \dots y],$$

and

$$\sigma\left\langle x, y \right\rangle = \sum_{[x \dots y] \in \langle x, y \rangle} \sigma[x \dots y],$$

where $\sigma[x \dots y]$ can be replaced with the equivalent terms given in Eq. 2.

The above definitions are illustrated by means of a digital image consisting of three different features, namely, *red, green* and *blue* shown in Fig.1. Inputs to this normalized flow graph are nodes r_1, r_2 and r_3, whereas the outputs are nodes b_1, b_2 and b_3. Nodes g_1, g_2 and g_3 represent the internal

nodes of the normalized flow graph.

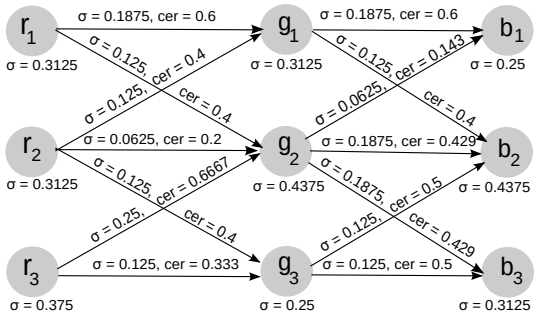

Figure 1: Normalized flow graph $\mathcal{G}$.

3 Related Works

Relationship between rough sets and flow graphs is discussed in [3] where the basic definitions of rough sets such as approximations, vagueness, accuracy of approximations and dependency degree are defined in terms of flow graphs if the initial data is in the form of flow graph. Layers of the flow graph depict the partition of universe with nodes representing the subsets of the universe. The branches of the flow graphs carrying coefficients are used in performing rough set approximation. Information flow in a flow graph is governed by Bayesian rule. In [28, 26, 25], certainty and coverage coefficients of the flow graph satisfy the Bayesian rule without referring to prior and posterior probabilities.

Relationship between decision trees and flow graphs is discussed in [26, 31]. A flow graph can be produced from the decision tree by eliminating the root node. The attributes of the flow graphs form nodes of the resultant decision tree. When the coefficients of paths are substituted for the complete path, a fusion flow graph is generated that reveals the overall structure of a flow graph in a simplified manner. Also, in comparison to the decision trees, flow graphs provide better insight and understanding of the data structure. Moreover, if the given information system is the decision table, then the last layer represents the decision attribute, whereas the other layers represent the conditional attributes. This structure of the flow graph helps in generating the decision rules for a given data set. The nodes of the flow graph in such situation serve as a logical formula, where the value of the node say $\sigma(x)$ is interpreted as a truth value [16] where $\langle 0, 1\rangle$ i.e. $0 < \sigma(x) < 1$. Thus $\sigma(x)$ can be understood as flow distribution ratio, probability or a truth value. With every branch (x, y) a rule is generated i.e. if x then y where x is a condition and y is a decision. If this sequence is generated for all the paths in the flow graph then the resultant set of rules together will form a decision algorithm for a given flow graph.

In [20] quality and predictability of flow graphs are defined in terms of entropy and information

gain respectively. The paper proposes the definitions for both the joint and conditional entropy in terms of the throughflow of the flow graph. Entropy of the overall flow graph is calculated, which is used in the information gain computation for the conditional attributes. The results of information gain for different conditional attributes are compared to retrieve the attribute with high predictive power. Also, it is shown that the layers of the flow graphs play a vital role in determining the overall performance and structure of the flow graph. Since a flow graph can be generated from the decision tree by eliminating the root node, the inverse order cannot be applied as the layers of flow graphs can be rearranged any number of times. Hence, with every new arrangement, a new decision tree can be formed. Moreover, while building the decision tree, the order of attributes with high information gain is taken into consideration i.e. the attribute with highest information gain forms the first layer and other layers are formed in a similar manner.

4 Perceptual Flow Graphs

Near set theory introduced in [33, 34] grew out of rough set theory [22, 23] and by the work of E.Orłowska on approximation spaces [18, 19]. Disjoint sets containing objects with similar descriptions are near sets. Similarity is determined quantitatively via some description of the objects. Near set theory is characterized by a perceptual system, whose objects are associated by relations such as indiscernibility, weak indiscernibility, and tolerance, taking into account descriptions using a tuple of probe functions [9] which provides a formal basis for identifying, comparing and measuring resemblance of objects based upon these descriptions [10]. Near sets are considered as a generalization of rough sets [44]. The notion of nearness not only differs from indiscernibility but is a more general concept and in consequence all basic notions of rough sets can be obtained within the near set framework [45]. The principal difference between rough set theory and near sets is that near sets can be discovered without the approximation of sets [38]. The theory of near sets can be summarized in three simple concepts: a perceptual system, a nearness relation and a near set [46]. The CBIR task can be considered as measuring nearness between two disjoint perceptual systems (i.e., two digital images).

4.1 Formal Model

We now present the formal model of flow graphs induced by perceptual systems. Note that a perceptual system $\langle O, \mathbb{F} \rangle$, where $\mathbb{F} = \phi_1, \phi_2, \ldots \phi_n$ is a finite set probe functions, gives rise to a directed acyclic finite graph $\mathcal{G} = (N,\ B)$ [40]. Each probe function ϕ_i induces a set of equivalence classes which will serve as nodes of $\mathcal{G}$: x and y belongs to the same equivalence class of ϕ_i provided

that $\phi_i(x) = \phi_i(y)$. A partition P_{ϕ_i} induced by ϕ_i will be called i-th layer. Then N is a disjoint union of all P_{ϕ_i}. The pair of nodes $([x]_{\phi_i}, [y]_{\phi_j})$ is a directed branch (that is an element of B) provided that their intersection is non empty and $j = i + 1$. By intersection we mean, that we seek objects that share the same values for given probe functions. Note also, that a directed acyclic finite graph induced by a perceptual system has some specific feature: it posses a linearly ordered set of layers (the order of probe functions determines the order of layers).

4.2 Normalized Flow Graphs with Layers

In what follows, due to area of application (that is image analysis), we are concerned mainly with normalized flow graphs whose underlying graphs are induced by perceptual systems. Following above remarks we shall call these graphs *normalized flow graphs with layers*, but in the explicit context of a perceptual system we shall refer to them also as *perceptual normalized flow graphs*.

Definition 12 ([40]) *Let $\mathcal{G}$ be a normalized flow graph (see Def. 11) with n layers and i^{th} layer having k_i nodes that are $x_i^1, x_i^2, \ldots, x_i^{j_i}, \ldots, x_i^{k_i}$, $1 \leq i \leq n$, $1 \leq j_i \leq k_i$ where the distance between node $x_l^{j_l}$ and $x_m^{j_m}$ can be calculated for both connected and disconnected nodes using as follows:*

$$\rho\left(x_l^{j_l}, x_m^{j_m}\right) = \begin{cases} \sigma\left(x_l^{j_l}\right) + \sigma\left(x_m^{j_m}\right) - 2\sigma\left\langle x_l^{j_l}, x_m^{j_m}\right\rangle & \text{, if C1} \\ \infty & \text{, if C2} \end{cases}$$

where C1 is the condition $x_l^{j_l}$ and $x_m^{j_m}$ are connected nodes, C2 is the condition $x_l^{j_l}$ and $x_m^{j_m}$ are disconnected nodes, $1 \leq l < m \leq n$, $1 \leq j_l \leq k_l$ and $1 \leq j_m \leq k_m$.

Note, that in the above calculation of the distance, the connected nodes are not necessarily input and output nodes. Recall that a connection between input and output node is called a *complete connection*. The connection between node $x_l^{j_l}$ and node $x_m^{j_m}$ can be calculated using

$$\sigma\left\langle x_l^{j_l}, x_m^{j_m}\right\rangle = \sum_{[x_l^{j_l} \ldots x_m^{j_m}] \in \left\langle x_l^{j_l}, x_m^{j_m}\right\rangle} \sigma[x_l^{j_l} \ldots x_m^{j_m}],$$

where

$$\sigma[x_l^{j_l} \ldots x_m^{j_m}] = \sigma(x_l^{j_l}) cer[x_l^{j_l} \ldots x_m^{j_m}],$$

and

$$cer[x_l^{j_l} \ldots x_m^{j_m}] = \prod_{i=l}^{m-1} cer(x_i^{j_i}, x_{i+1}^{j_{i+1}}).$$

We now introduce the formalism for establishing nearness between two normalized flow graphs $\mathcal{G}$ and $\mathcal{G}'$ necessary of comparing digital images.

Definition 13 ([40]) *Let $\mathcal{G} = (N, B, \varphi, \sigma)$ be a normalized flow graph with n layers and the i^{th} layer having k_i nodes where $1 \leq i \leq n$, $1 \leq j_i \leq k_i$: $x_i^1, x_i^2, \ldots, x_i^{j_i}, \ldots, x_i^{k_i}$. Let $\mathcal{G}' = (N', B', \varphi', \sigma')$ be a normalized flow graph with n' layers and the i'^{th} layer having k_i' nodes: $x_i'^1, x_i'^2, \ldots, x_i'^{j_i}, \ldots, x_i'^{k_i}$. The distance between path $\left[x_1^{j_1} x_2^{j_2} \ldots x_{j_1}^{j_n}\right]$ in $\mathcal{G}$ and path $\left[x_1'^{j_1} x_2'^{j_2} \ldots x_{j_1}'^{j_n}\right]$ in $\mathcal{G}'$ such that node $x_i^{j_i}$ in $\mathcal{G}$ and node $x_i'^{j_i}$ in $\mathcal{G}'$, $1 \leq i \leq n$, represent probe functions for the same feature is defined as*

$$\rho_p\left(\left[x_1^{j_1} x_2^{j_2} \ldots x_{j_1}^{j_n}\right], \left[x_1'^{j_1} x_2'^{j_2} \ldots x_{j_1}'^{j_n}\right]\right) = \frac{\left|\sigma\left[x_1^{j_1} x_2^{j_2} \ldots x_{j_1}^{j_n}\right] - \sigma'\left[x_1'^{j_1} x_2'^{j_2} \ldots x_{j_1}'^{j_n}\right]\right|}{max\left\{\sigma\left[x_1^{j_1} x_2^{j_2} \ldots x_{j_1}^{j_n}\right], \sigma'\left[x_1'^{j_1} x_2'^{j_2} \ldots x_{j_1}'^{j_n}\right]\right\}}.$$

Next, the distance between a set of paths of two flow graphs $\mathcal{G}$ and $\mathcal{G}'$ is given in the following definition.

Definition 14 ([40]) *Let $\mathcal{G} = (N, B, \varphi, \sigma)$ be a normalized flow graph with n layers and the i^{th} layer having k_i nodes where $1 \leq i \leq n$, $1 \leq j_i \leq k_i$: $x_i^1, x_i^2, \ldots, x_i^{j_i}, \ldots, x_i^{k_i}$. Let $\mathcal{G}' = (mN', B', \varphi', \sigma')$ be a normalized flow graph with n' layers and the i'^{th} layer having k_i' nodes: $x_i'^1, x_i'^2, \ldots, x_i'^{j_i}, \ldots, x_i'^{k_i}$. The distance between a set of paths of two normalized flow graphs $\mathcal{G}$ and $\mathcal{G}'$ is defined by*

$$D_{\rho_p}(\mathcal{G}, \mathcal{G}') = \begin{cases} \inf\left\{\rho_p\left(p, p'\right) : p \in P, p' \in P'\right\} & , \textit{if C3}, \\ \infty & , \textit{if C4}, \end{cases}$$

where P is the set of all the paths from $\mathcal{G}$, P' is the set of all the paths from $\mathcal{G}'$, C3 is the condition $P \neq \emptyset$ and $P' \neq \emptyset$, and C4 is the condition $P = \emptyset$ or $P' = \emptyset$.

In practical applications, determining the set of all possible paths among given input and output nodes of the normalized flow graph results in lengthy execution time. So, in order to overcome this problem, Def. 14 has been *modified*. Instead of searching for all the possible paths among input and output nodes of the normalized flow graph, we only focus on the set of shortest paths among input and output nodes of the normalized flow graph. The set of shortest paths is labelled SP.

So, the *new definition* to find distance between a set of paths of two normalized flow graphs $\mathcal{G}$ and $\mathcal{G}'$ is defined as

Definition 15 *Let $\mathcal{G} = (N, B, \varphi, \sigma)$ be a normalized flow graph with n layers and the i^{th} layer having k_i nodes where $1 \leq i \leq n$, $1 \leq j_i \leq k_i$: $x_i^1, x_i^2, \ldots, x_i^{j_i}, \ldots, x_i^{k_i}$. Let $\mathcal{G}' = (N', B', \varphi', \sigma')$ be a normalized flow graph with n' layers and the i'^{th} layer having k_i' nodes: $x_i'^1, x_i'^2, \ldots, x_i'^{j_i}, \ldots, x_i'^{k_i}$.*

$$D_{\rho_p}(\mathcal{G}, \mathcal{G}') = \begin{cases} \inf\left\{\rho_p\left(p, p'\right) : p \in SP, p' \in SP'\right\} & , \textit{if C3}, \\ \infty & , \textit{if C4}, \end{cases}$$

where SP is the set of all the shortest paths from $\mathcal{G}$, SP' is the set of all the shortest paths from $\mathcal{G}'$,

C3 is the condition $SP \neq \emptyset$ and $SP' \neq \emptyset$, and C4 is the condition $SP = \emptyset$ or $SP' = \emptyset$.

Example:

We now return to Fig.1, to illustrate an example of shortest distance computation between node r_1 and node b_2. First, we calculate $\sigma\langle r_1, b_2\rangle$ as follows:

$$\begin{aligned}
\sigma\langle r_1, b_2\rangle &= \inf\{\sigma[r_1 g_1 b_2], \sigma[r_1 g_2 b_2]\},\\
&= \inf\{\sigma(r_1) cer[r_1 g_1 b_2], \sigma(r_1) cer[r_1 g_2 b_2]\},\\
&= \inf\{\sigma(r_1)\, cer(r_1, g_1)\, cer(g_1, b_2),\\
&\quad \sigma(r_1)\, cer(r_1, g_2)\, cer(g_2, b_2)\},\\
&= \inf\{(0.3125)(0.6)(0.4), (0.3125)(0.4)\\
&\quad (0.429)\},\\
&= \inf\{0.075, 0.053625\} = \{0.053625\}.
\end{aligned}$$

Then, by Def. 12, we have

$$\rho(r_1, b_2) = \sigma(r_1) + \sigma(b_2) - 2\sigma\langle r_1, b_2\rangle = 0.3125 + 0.4375 - 2(0.053625) = 0.64275.$$

By Def. 15, the distance between set of shortest paths in Figure 2 and 3 is computed as:

$$\begin{aligned}
\rho([r_1 b_1], [r_1', b_1']) &= \frac{|\inf\{\sigma[r_1 g_1 b_1], \sigma[r_1 g_2 b_1]\} - \inf\{\sigma[r_1', g_1', b_1']\}|}{\max\{\inf\{\sigma[r_1 g_1 b_1], \sigma[r_1 g_2 b_1]\}, \inf\{\sigma[r_1', g_1', b_1']\}\}},\\
&= \Big\{|\inf\{(0.3125)(0.6)(0.6), (0.3125)(0.4)(0.143)\}\\
&\quad - \inf\{(0.375)(0.333)(0.5)\}|\Big\}\\
&\quad \div \max\Big\{\inf\{(0.3125)(0.6)(0.6),\\
&\quad (0.3125)(0.4)(0.143)\}, \inf\{(0.375)(0.333)(0.5)\}\Big\},\\
&= \frac{|\inf\{0.1125, 0.017875\} - \inf\{0.06243\}|}{\max\{\inf\{0.1125, 0.017875\}, \inf\{0.06243\}\}},\\
&= \frac{|0.017875 - 0.06243|}{\max\{0.017875, 0.06243\}},\\
&= 0.7136,
\end{aligned}$$

$$\vdots$$

$$
\begin{aligned}
\rho\left([r_3b_2],[r'_3,b'_2]\right) &= \\
&\Big\{\left|\{\inf\{\sigma\left[r_3g_2b_2\right],\sigma\left[r_3g_3b_2\right]\}\}-\inf\{\{\sigma\left[r'_3g'_2b'_2\right],\sigma\left[r'_3g'_3b'_2\right]\}\,|\}\right\} \\
&\div\max\Big\{\inf\left\{\sigma\left[r_3g_2b_2\right],\sigma\left[r_3g_3b_2\right]\right\},\inf\left\{\sigma\left[r'_3g'_2b'_2\right],\sigma\left[r'_3g'_3b'_2\right]\right\}\Big\}, \\
&= \Big\{|\inf\{(0.375)(0.6667)(0.429),(0.375)(0.333)(0.5)\} \\
&\quad -\inf\{(0.375)(0.333)(0.333),(0.375)(0.667)(0.333)\}|\Big\} \\
&\quad \div\max\Big\{\inf\{0.375)(0.6667)(0.429),(0.375)(0.333)(0.5)\}, \\
&\quad \inf\{(0.375)(0.333)(0.5),(0.375)(0.667)(0.333)\}\Big\}, \\
&= |\inf\{0.107255,0.0624375\}-\inf\{0.04158,0.08329\}| \\
&\quad \div\max\Big\{\inf\{0.107255,0.0624375\},\inf\{0.04158,0.08329\}\Big\}, \\
&= \frac{|0.0624375-0.04158|}{\max\{0.0624375,0.04158\}}=0.334.
\end{aligned}
$$

Next, consider the distance between set of paths of $\mathcal{G}$ and $\mathcal{G}'$, then by Def. 15, we have

$$
D_{\rho_p}(\mathcal{G},\mathcal{G}')=\inf\{\rho\left([r_1b_1],[r'_1b'_1]\right),\ldots,\rho\left([r_3b_2],[r'_3b'_2]\right)\}=\inf\{0.7136,\ldots,0.334\}=0.334.
$$

The degree of nearness between two normalized flow graphs depends upon the value of distance computation performed above. Two images are completely near to each other, if, the result from the above computation is zero. In case of non-zero value, closer the resultant value to the zero, more similar(near) the images are.

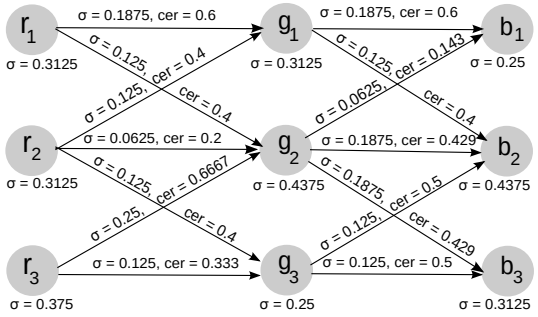

Figure 2: Normalized flow graph $\mathcal{G}$.

4.3 Application to CBIR

The PFG algorithm is applied to a CBIR problem, where the goal is to retrieve digital images from databases based on the content of an image rather than on some semantic string or keywords asso-

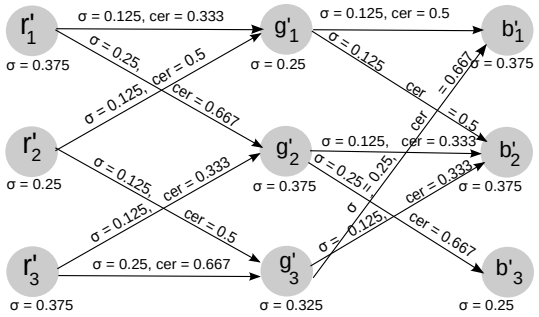

Figure 3: Normalized flow graph $\mathcal{G}'$.

ciated with the image. The content of the image is determined by probe functions that characterize features such as colour, texture, shape of objects, and edges. In our approach to CBIR, a search entails analysis of content, based on a nearness measure between a query image and a test image. To generate results, the SIMPLIcity image database [43], a database of images containing 10 categories with 100 images in each category, was used (see, *e.g.*, Fig. 4). The categories are varied with different objects and scenes, and images in different categories can also resemble each other. This dataset was selected to provide a basis for comparison with the tolerance nearness measure results reported in [11] (see also [10]).

Figure 4: Sample images from SIMPLIcity dataset.

The results were generated by partitioning the images into subimages, where each subimage is considered an object in the near set sense, *i.e.* each subimage is a perceptual object, and each object description consists of the values obtained from image-based probe functions applied to the subimage. This technique of partitioning an image and assigning feature vectors (*i.e.* object descriptions) to each subimage is an approach that has also been traditionally used in CBIR.

The results in this article are obtained using the same test data for the results reported in [11]. In particular, each subimage is of size 20×20 (resulting in 456 objects per image pair). In [11], images were characterized by 18 features (obtained via probe functions), namely 4 texture features obtained from the grey-level co-occurrence matrix of subimage, the first and second moments of u and v in the CIELUV colour space, the number of edge pixels contained in the subimage based on Mallat's multiscale edge detection method [17], and the Zernike moments of order 4, excluding $\tilde{A}_{00}$. However, due to large runtimes, a reduced set of features were used to obtain results.

Specifically, the machine learning tool, Weka [7], was used to perform feature reduction among the 18 features listed above. Weka aims to provide a comprehensive collection of machine learning algorithms and data pre-processing tools, thereby allowing users to quickly experiment with machine learning algorithms on new data sets. The goal of feature reduction is to select the best subset of features from the given list that is necessary and sufficient to describe the target concept [39]. Feature reduction can also be performed from rough set based methods such as reduct and core [8]. However for this work, the gain ratio algorithm [13] was selected to perform feature reduction on data reported in [11]. The top four features selected after performing the evaluation were: the number of edge pixels, Zernike moment of order 11, the average U and average V value from the CIELUV colour space.

Algorithm 1 details the approach to using normalized flow graphs for CBIR. First, the input to the algorithm is a dataset containing the output of real-valued probe functions obtained by processing all images in a given CBIR image database. Next, the output is a matrix quantifying the similarity of each pair of images in the database. Specifically, for a given a, b (representing two images in the database), two graphs, $\mathcal{G}, \mathcal{G}'$, are created. Then, Defn. 15 is used to populate row a and column b of the matrix. These matrix values are used to rank the results for a given query image. For example, row a of the matrix can be sorted (from lowest to highest), and the order of the sorted values represent the retrieval results for query image a. In other words, the sorted distance measures represent the retrieved images from the database relevant to query image a (from best to worst).

Next, the body of Algorithm 1 summarizes the approach to using Defn. 15 for CBIR, *i.e.* it presents the approach for determining the degree of nearness between two perceptual normalized

Algorithm 1: Perceptual Flow Graph Algorithm

Input : Processed image dataset (*i.e.* probe functions output for all images in database).
Output: Matrix, M, populated by Defn. 15.

for each probe function ϕ_i **do**
  Discretize all output for ϕ_i into n bins.
for a in database **do**
  Generate flow graph $\mathcal{G}$ for image a.
  for b in database **do**
    Generate flow graph $\mathcal{G}'$ for image b.
    Quantify the degree of similarity between flow graphs $\mathcal{G}$ and $\mathcal{G}'$ *s.t.* $D_{\rho_p}(\mathcal{G}, \mathcal{G}') =$

$$\begin{cases} \inf\left\{\rho_p\left(\left[x_1^{j_1}\ldots x_{j_1}^{j_n}\right], \left[x_1^{\prime j_1}\ldots x_{j_1}^{\prime j_n}\right]\right) : \left[x_1^{j_1}\ldots x_{j_1}^{j_n}\right] \in SP, \left[x_1^{\prime j_1}\ldots x_{j_1}^{\prime j_n}\right] \in SP'\right\}; \\ \qquad\qquad \text{if } SP \neq \emptyset \text{ and } SP' \neq \emptyset, \\ \infty; \qquad\qquad \text{if } SP = \emptyset \text{ or } SP' = \emptyset, \end{cases}$$

    where SP and SP' denote set of shortest paths in $\mathcal{G}$ and $\mathcal{G}'$, respectively.
    Populate row a and column b of M with $D_{\rho_p}(\mathcal{G}, \mathcal{G}')$.
    Populate row b and column a of M with $D_{\rho_p}(\mathcal{G}, \mathcal{G}')$ (unless $a == b$).

flow graphs for use in CBIR. First, the probe function input is discretized into n bins to reduce algorithm runtime, where $n \in \{5, 10, 15, 20, 25, 30, 35\}$. Second, for a given a, b, two graphs, $\mathcal{G}, \mathcal{G}'$, are constructed with the following order of probe functions: Avg. V value from CIELUV, Zernike moment of order 11, number of edge pixels and Avg. U value from CIELUV. Layers of the flow graph can be rearranged any number of times, and with every different arrangement, a new result is obtained [20]. The order of layers mentioned above was selected because it produced the best results for all possible permutations of the four probe. Next, Dijkstra's Shortest Path Algorithm [4] was required to implement Defn. 15. Recall, this algorithm performs addition on edge weights as traversal proceeds along the paths of the graph. However, multiplication was used in this implementation to reflect the calculation of certainty for a path (see Eq. 1). Finally, as was mentioned, the approach presented here is compared with the tolerance nearness measure [11].

5 Tolerance Nearness Measure

Tolerance nearness measure quantifies the similarity of two images as follows. First, the subimages represent objects in a perceptual system, *i.e*, let the sets X and Y represent the two images to be compared where each set consists of the subimages obtained by partitioning the images. Furthermore, the set of all objects in this perceptual system is given by $Z = X \cup Y$. Then, sets called tolerance classes are obtained from Z, where each pair of objects in the set a tolerance relation and the set is maximal with respect to inclusion. Next, the nearness of X and Y is determined by the distribution of the tolerance classes between the two sets X and Y. The idea is that tolerance classes

obtained from images containing similar content (based on the selected probe functions) should be evenly divided between the two sets X and Y.

6 Results and Discussion

This section presents CBIR results using Algorithm 1 and compares them with those reported with the tolerance nearness measure. The results are presented using precision *vs* recall plots, where the idea is to retrieve all images from the same category as the query image before images from any other category. Each image in the SIMPLIcity database was compared with each other image, and the results are sorted in the ascending order. Typically, the smallest value represents the "nearest" image, which typically is the query image itself. Similarly, in the ideal case, all images from the same category would be retrieved before any images from other categories. In this case, precision would be 100% until recall reached 100%, at which point precision would drop to the number of images in query category / number of images in the database. As a result, our final value of precision will be $\sim$11% since we used 9 categories each containing 100 images. Note, only 9 categories were used since the category 4 (cartoons of dinosaurs) are easy to retrieve and their inclusion would only increase the runtime of the experiment. The results for PFG are presented in Fig. 5 - 8, where the PFG average precision vs. recall plots are given in Fig. 5 and 7, and the PFG best precision vs. recall results are given in Fig. 6 and 8. These plots were generated for bin numbers $n \in \{5, 10, 15, 20, 25, 30, 35\}$. On the other hand, the results for tNM are shown in Fig. 9 - 12, where the tNM average precision vs. recall plots are given in Fig. 9 and 11, and the tNM best precision vs. recall results are given in Fig. 10 and 12.

Next, the following discusses some observations of the reported results. First, notice that some of the curves have a sharp point of inflection (see, *e.g.*, bin no = 35 at 28% recall in Fig. 5(a) and for $\varepsilon = 0.1$ at 18% in Fig. 9(a)). These points represent the location at which, on average, the distance measure values for a particular query image and the remaining images become infinite (*i.e.*, the measure indicates these images are not near each other). In order to consistently provide this clear demarcation, any images from the same category as the query image that produced an infinite distance measure were ranked last in the search, thus giving this same feature in all the plots.

The results for the PFG best plots are reported in Fig. 6 and 8. The drop at 78% recall for bin no = 15 in Fig.6(a) signifies that 78 images are retrieved accurately from the correct category before an image from the category different to the query image category is encountered. It must be noticed that number of images to be retrieved correctly depends entirely on the bin number. In some cases, lower bin numbers retrieve more number of images from the exact category than the higher bin numbers. As in Fig. 6(c), 6(e), 6(f), 8(a), 8(b), 8(c), lower bin numbers (5) fetch more

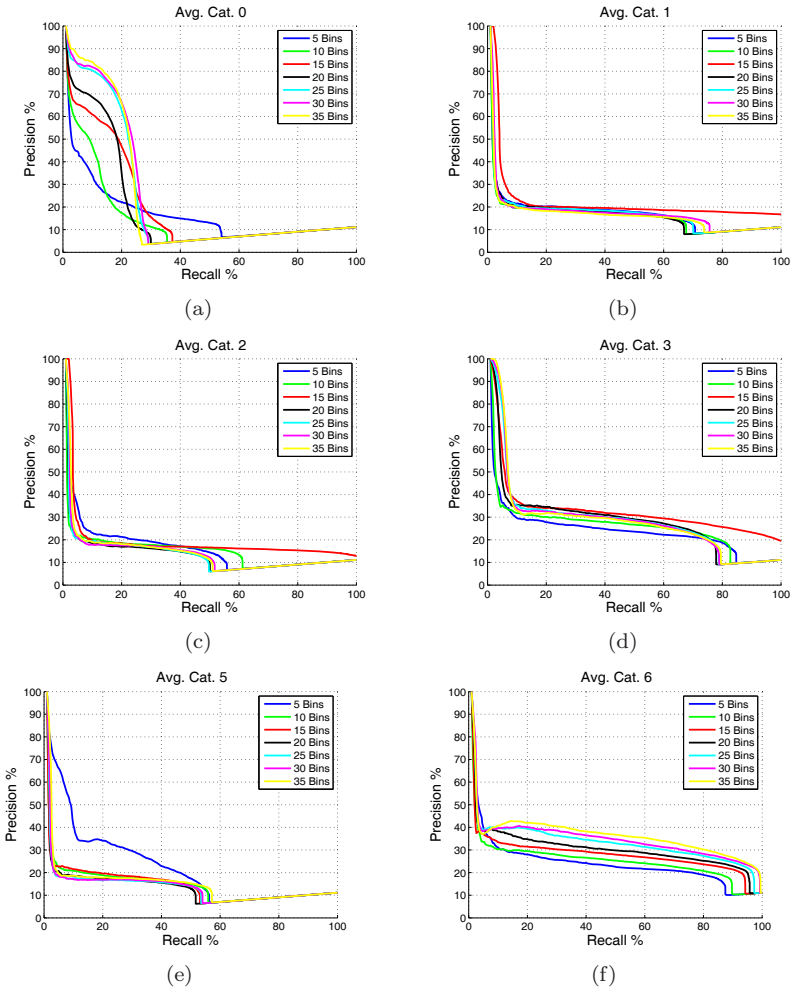

Figure 5: PFG Average precision versus recall plots grouped by category: (a) - (f) Cat. 0 - 6 (excluding cat. 4).

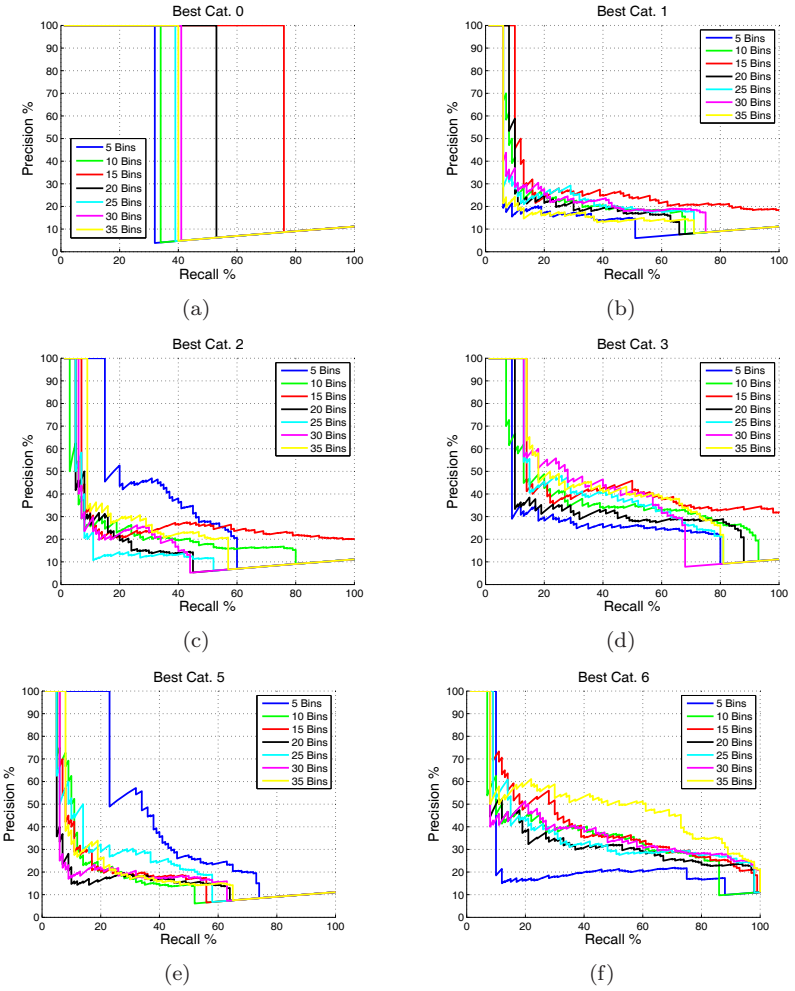

Figure 6: PFG Best precision versus recall plots grouped by category: (a) - (f) Cat. 0 - 6 (excluding cat. 4).

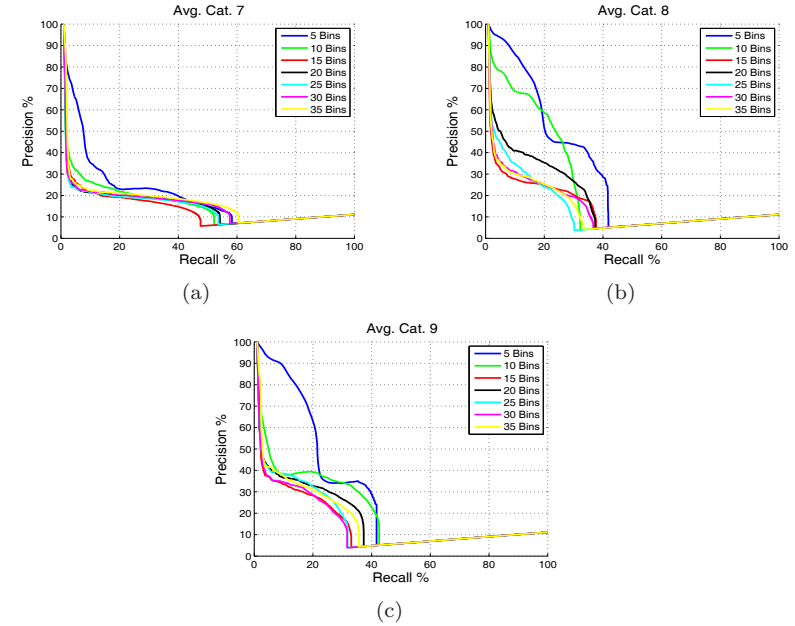

Figure 7: PFG Average precision versus recall plots grouped by category: (a) - (c) Cat. 7 - 9.

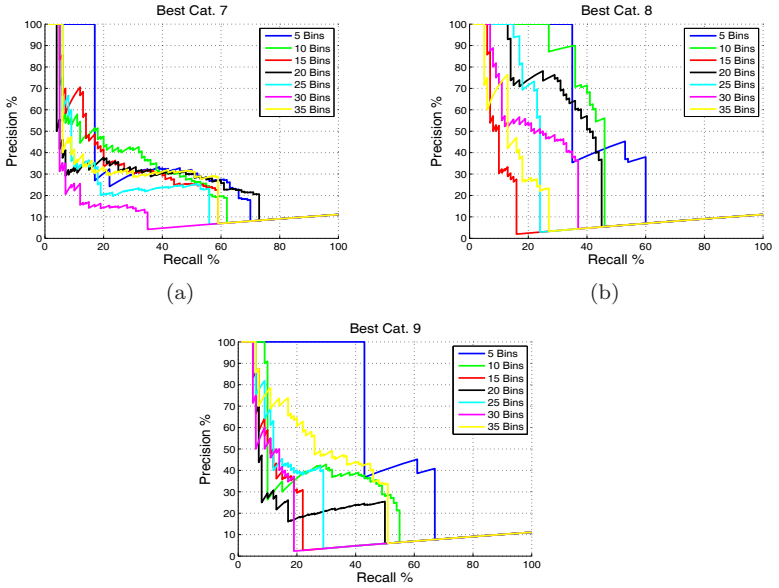

Figure 8: PFG Best precision versus recall plots grouped by category: (a) - (c) Cat. 7 - 9.

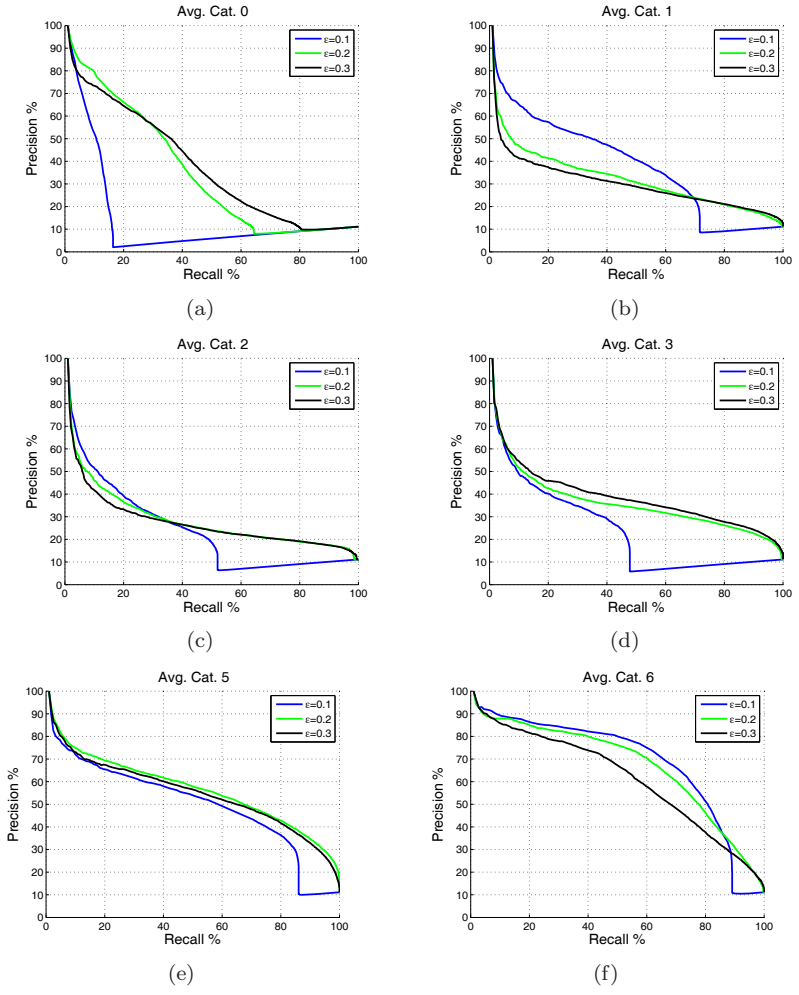

Figure 9: tNM Average precision versus recall plots grouped by category: (a) - (f) Cat. 0 - 6 (excluding cat. 4).

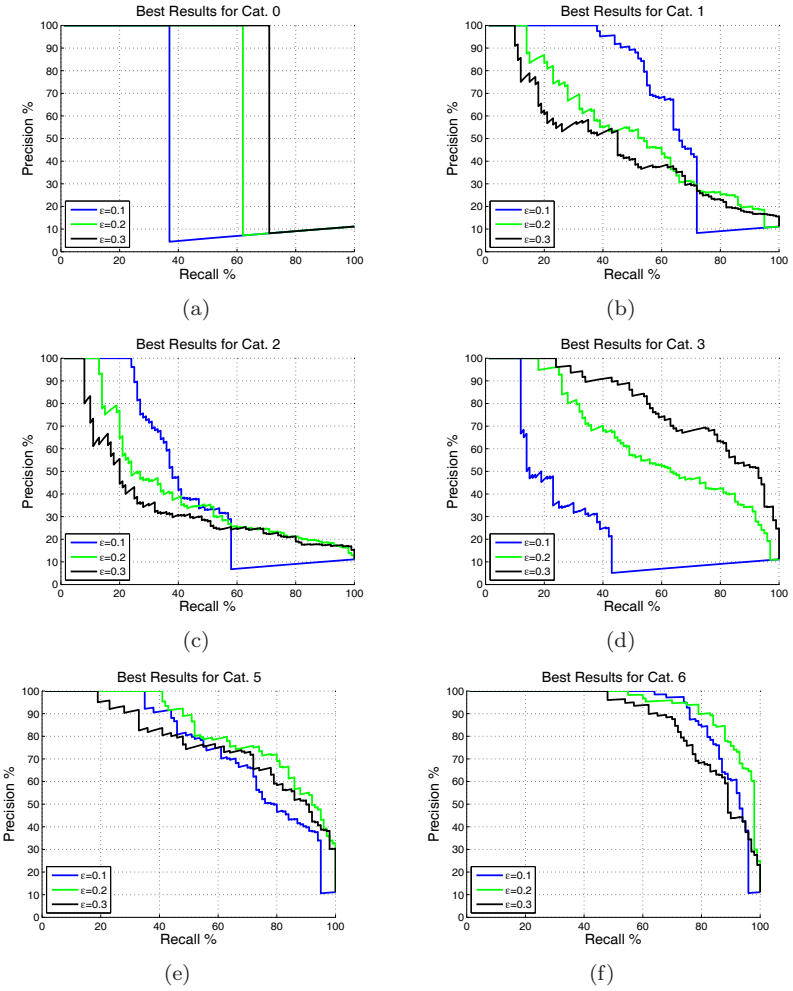

Figure 10: tNM Best precision versus recall plots grouped by category: (a) - (f) Cat. 0 - 6 (excluding cat. 4).

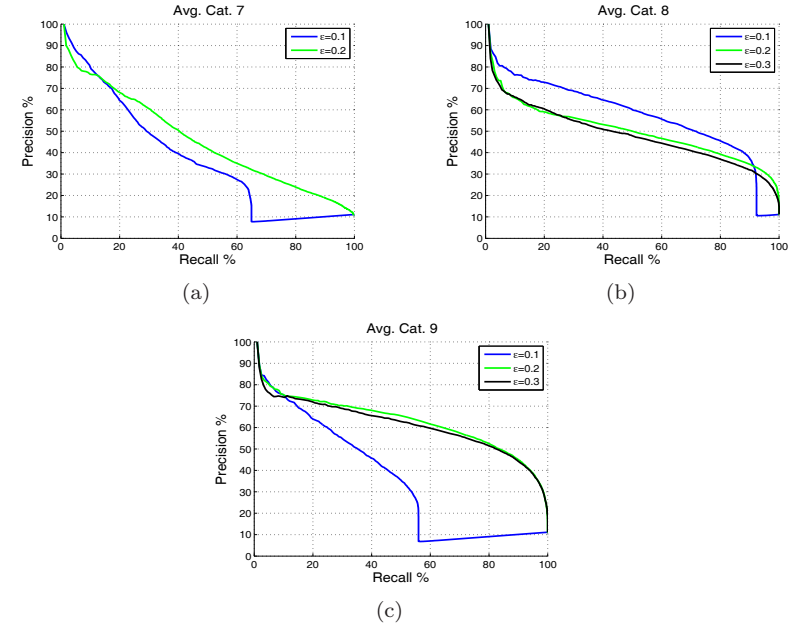

Figure 11: tNM Average precision versus recall plots grouped by category: (a) - (c) Cat. 7 - 9.

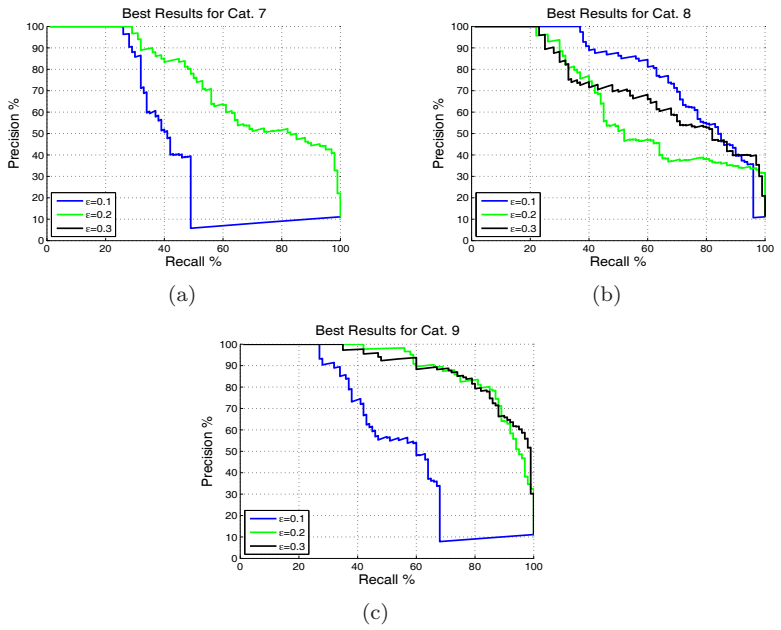

Figure 12: tNM Best precision versus recall plots grouped by category: (a) - (c) Cat. 7 - 9.

number of images from the same category as the query image, while in 6(d), higher bin number (35), retrieves more number of images from the correct category. Comparing PFG results with tNM, it was observed that for the tribal category, PFG fetched more number of images from the database than tNM, whereas for the category of beaches, buses, elephants, flowers and horses, tNM showed higher retrieval of accurate images. On the other hand, for buildings, mountains and food categories, both PFG and tNM had comparable retrieval of images from the databases. Overall, it was observed that PFG was able to generate comparable results with tNM.

7 Conclusion

The contribution of this paper is a practical implementation of perceptual flow graph (PFG) algorithm which uses a perceptual indiscernibility relation from near sets and data pre-processing strategy, binning, to perform CBIR on digital images. Perceptually relevant information was extracted from a set objects formed from pairs of images, where each object has an associated object description. Overall, PFG is able to demonstrate comparable performance with tolerance nearness measure (tNM) measure in terms of precision and recall on the SIMPLIcity image database which is a repository of images containing 10 categories with 100 images in each category. An important characteristic of the perceptual flow graphs are probe functions that characterize features. Selection of probe functions and the order of their representation in a perceptual flow graph are the two key steps involved in computing nearness. Future work includes experiments with other image data sets for different combination of probe functions and their ordering. Furthermore, the theory of flow

graphs can be extended by introducing tolerance to the nearness measure computation. Another application of the PFG algorithm is to consider multimedia data sets.

Acknowledgements

This research has been supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) discovery grants 194376 and 418413.

References

[1] C. J. Butz, W. Yan, and B. Yang. An efficient algorithm for inference in rough set flow graphs. In *Transactions on Rough Sets V*, pages 102–122. Springer, 2006.

[2] D. Chitchareon and P. Pattaraintakorn. Knowledge discovery by rough sets mathematical flow graphs and its extension. In *Proceedings of the 26th IASTED International Conference on Artificial Intelligence and Applications*, AIA '08, pages 340–345. ACTA Press, 2008.

[3] D. Chitcharoen and P. Pattaraintakorn. An extension of rough set approximation to flow graph based data analysis. In *Rough Sets and Current Trends in Computing*, volume 6086 of *Lecture Notes in Computer Science*, pages 418–427. Springer Berlin Heidelberg, 2010.

[4] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. The MIT Press, Cambridge, Massachusetts, 1990.

[5] A. Czyzewski and B. Kostek. Musical metadata retrieval with flow graphs. In *Rough Sets and Current Trends in Computing*, volume 3066 of *Lecture Notes in Computer Science*, pages 691–698. 2004.

[6] S. Greco, Z. Pawlak, and R. Słowiński. Generalized decision algorithms, rough inference rules, and flow graphs. In *Rough Sets and Current Trends in Computing*, volume 2475 of *Lecture Notes in Computer Science*, pages 93–104. Springer Berlin Heidelberg, 2002.

[7] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[8] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, August 2000.

[9] C. J. Henry. *Near Sets: Theory and Applications*. PhD thesis, University of Manitoba, 2011.

[10] C. J. Henry. Perceptual indiscernibility, rough sets, descriptively near sets, and image analysis. In James F. Peters and Andrzej Skowron, editors, *Transactions on Rough Sets XV*, volume 7255 of *Lecture Notes in Computer Science*, pages 41–121. Springer Berlin Heidelberg, 2012.

[11] C. J. Henry and S. Ramanna. Maximal clique enumeration in finding near neighbourhoods. In *Transactions on Rough Sets XVI*, volume 7736 of *Lecture Notes in Computer Science*, pages 103–124. Springer Berlin Heidelberg, 2013.

[12] C. J. Henry and S. Ramanna. Signature-based perceptual nearness: Application of near sets to image retrieval. *Mathematics in Computer Science*, 7(1):71–85, 2013.

[13] A. G. Karegowda, A. S. Manjunath, and M. A. Jayaram. Comparative study of attribute selection using gain ratio and correlation based feature selection. *International Journal of Information Technology and Knowledge Management*, 2(2):271–277, 2010.

[14] B. Kostek and A. Czyzewski. Processing of musical metadata employing pawlaks flow graphs. In *Transactions on Rough Sets I*, volume 3100 of *Lecture Notes in Computer Science*, pages 279–298. Springer Berlin Heidelberg, 2004.

[15] H. Liu, J. Sun, and H. Zhang. Interpretation of extended pawlak flow graphs using granular computing. In *Transactions on Rough Sets VIII*, volume 5084 of *Lecture Notes in Computer Science*, pages 93–115. Springer Berlin Heidelberg, 2008.

[16] J. Lukasiewicz. Die logishen grundlagen der wahrscheinilchk eitsrec hmg. krakow(1913). *L. Borkowski (ed.), Jan Lukasiewics - Selected Works,North Holland Publishing Company, Amsterdam, Polish Scientific Publishers, London, Warsa w 16-63*, 1970.

[17] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, California, 1999.

[18] E. Orłowska. Semantics of vague concepts. applications of rough sets. In *Tech. Rep. 469, Institute for Computer Science*. Polish Academy of Sciences, 1982.

[19] E. Orłowska. Semantics of vague concepts. In Georg Dorn and Paul Weingartner, editors, *Foundations of Logic and Linguistics. Problems and Solutions*, pages 465–482. Plenum Pres, London, 1985.

[20] P. Pattaraintakorn. Entropy measures of flow graphs with applications to decision trees. In *Rough Sets and Knowledge Technology*, volume 5589 of *Lecture Notes in Computer Science*, pages 618–625. Springer Berlin Heidelberg, 2009.

[21] P. Pattaraintakorn, N. Cercone, and K. Naruedomkul. Rule learning: Ordinal prediction based on rough sets and soft-computing. *Applied Mathematics Letters*, 19(12):1300 – 1307, 2006.

[22] Z. Pawlak. Classification of objects by means of attributes. *Institute for Computer Science, Polish Academy of Sciences*, pages 1–20, 1981.

[23] Z. Pawlak. Rough sets. In *International Journal of Computer and Information Sciences 11, 341-356*. 1982.

[24] Z. Pawlak. Decision algorithms, bayes theorem and flow graphs. In Leszek Rutkowski and Janusz Kacprzyk, editors, *Neural Networks and Soft Computing*, volume 19 of *Advances in Soft Computing*, pages 18–24. Physica-Verlag HD, 2003.

[25] Z. Pawlak. Decisions algorithms and flow graphs; a rough set approach. *Journal of Telecommunications and Information Technology*, pages 98–101, 2003.

[26] Z. Pawlak. Flow graphs and decision algorithms. In *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, volume 2639 of *Lecture Notes in Computer Science*, pages 1–10. Springer Berlin Heidelberg, 2003.

[27] Z. Pawlak. Probability, truth and flow graph. *Electronic Notes in Theoretical Computer Science*, 82(4):1 – 9, 2003. International Workshop on Rough Sets in Knowledge Discovery and Soft Computing (Satellite Event for {ETAPS} 2003).

[28] Z. Pawlak. Decisions rules and flow networks. *European Journal of Operational Research*, 154(1):184 – 190, 2004.

[29] Z. Pawlak. Flow graphs and data mining. *Trans. on Rough Sets*, III:1–36, 2005.

[30] Z. Pawlak. Rough sets and flow graphs. In *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, volume 3641 of *Lecture Notes in Computer Science*, pages 1–11. Springer Berlin Heidelberg, 2005.

[31] Z. Pawlak. Decision trees and flow graphs. In *Rough Sets and Current Trends in Computing*, volume 4259 of *Lecture Notes in Computer Science*, pages 1–11. Springer Berlin Heidelberg, 2006.

[32] J. F. Peters. Classification of objects by means of features. In *Proceedings of the IEEE Symposium Series on Foundations of Computational Intelligence(IEEE SCCI 2007)*, pages 1–8, 2007.

[33] J. F. Peters. Near sets. general theory about nearness of objects. *Applied Mathematical Sciences*, 1(53):2609–2629, 2007.

[34] J. F. Peters. Near sets. special theory about nearness of objects. *Fundam. Inf.*, 75(1-4):407–433, January 2007.

[35] J. F. Peters and D. Chitcharoen. Sufficiently near neighbourhoods of points in flow graphs. a near set approach. *Fundam. Inf.*, 124(1-2):175–196, January 2013.

[36] J. F. Peters, D. Chitcharoen, and S. Ramanna. Reasoning with near set-based digital image flow graphs. In *Multi-disciplinary Trends in Artificial Intelligence - 7th International Workshop, MIWAI 2013, Krabi, Thailand, December 9-11, 2013. Proceedings*, pages 199–210, 2013.

[37] J. F. Peters and S. Naimpally. S.: Applications of near sets. *Amer. Math. Soc. Notices*, 59(4):536–542, 2012.

[38] J. F. Peters and P. Wasilewski. Foundations of near sets. *Information Sciences*, 179(18):3091 – 3109, 2009.

[39] S. Piramuthu. Evaluating feature selection methods for learning in data mining applications. *European journal of operational research*, 156(2):483–494, 2004.

[40] S. Ramanna and D. Chitcharoen. Flow graphs: Analysis with near sets. *Mathematics in Computer Science*, 7(1):11–29, 2013.

[41] J. Sun, H. Liu, and H. Zhang. An extension of pawlaks flow graphs. In *Rough Sets and Knowledge Technology*, volume 4062 of *Lecture Notes in Computer Science*, pages 191–199. Springer Berlin Heidelberg, 2006.

[42] Z. Suraj and K. Pancerz. Flow graphs as a tool for mining prediction rules of changes of components in temporal information systems. In *Rough Sets and Knowledge Technology*, volume 4481 of *Lecture Notes in Computer Science*, pages 468–475. Springer Berlin Heidelberg, 2007.

[43] J. Z. Wang. Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1075–1088, 2003.

[44] M. Wolski. Perception and classification. a note on near sets and rough sets. *Fund. Inf.*, 101(1-2):143–155, January 2010.

[45] M. Wolski. Granular computing: Topological and categorical aspects of near and rough set approaches to granulation of knowledge. In *Transactions on Rough Sets XVI*, volume 7736 of *Lecture Notes in Computer Science*, pages 34–52. Springer Berlin Heidelberg, 2013.

[46] M. Wolski. Toward foundations of near sets: (pre-)sheaf theoretic approach. *Mathematics in Computer Science*, 7(1):125–136, 2013.