Knowledge and Information Systems (20XX) Vol. X: 1–29

Comparison of Descriptor Spaces for Chemical Compound Retrieval and Classification

Nikil Wale

Department of Computer Science,
University of Minnesota, Twin Cities.
Email: nwale@cs.umn.edu.

Ian A. Watson

Eli Lilly and Company.
Lilly Research Labs, Indianapolis
Email: watson_ian_a@lilly.com

George Karypis

Department of Computer Science,
University of Minnesota, Twin Cities
Email: karypis@cs.umn.edu.

Abstract. In recent years the development of computational techniques that build models to correctly assign chemical compounds to various classes or to retrieve potential drug-like compounds has been an active area of research. Many of the best-performing techniques for these tasks utilize a descriptor-based representation of the compound that captures various aspects of the underlying molecular graph's topology. In this paper we compare five different set of descriptors that are currently used for chemical compound classification. We also introduce four different descriptors derived from all connected fragments present in the molecular graphs primarily for the purpose of comparing them to the currently used descriptor spaces and analyzing what properties of descriptor spaces are helpful in providing effective representation for molecular graphs. In addition, we introduce an extension to existing vector-based kernel functions to take into account the length of the fragments present in the descriptors. We experimentally evaluate the performance of the previously introduced and the new descriptors in the context of SVM-based classification and ranked-retrieval on 28 classification and retrieval problems derived from 18 datasets. Our experiments show that for both of these tasks, two of the four descriptors introduced in this paper along with

the extended connectivity fingerprint based descriptors consistently and statistically outperform previously developed schemes based on the widely used fingerprint- and Maccs keys-based descriptors, as well as recently introduced descriptors obtained by mining and analyzing the structure of the molecular graphs.

Keywords: SVM; classification; retreival; descriptor space; kernel

1. Introduction

Discovery, design and development of new drugs is an expensive and challenging process. Any new drug should not only produce the desired response to the disease but should do so with minimal side effects. One of the key steps in the drug design process is the identification of the chemical compounds (*hit* compounds or just *hits*) that display the desired and reproducible behavior against the specific biomolecular target [33]. This represents a significant hurdle in the early stages of drug discovery. Therefore, computational techniques that build models to correctly assign chemical compounds to various classes or retrieve compounds of desired class from a database have become popular in the pharmaceutical industry.

Over the last twenty years extensive research has been carried out to identify representations of molecular graphs that can build good classification models or retrieve actives from a database in an effective way. Towards this goal, a number of different approaches have been developed that represent each compound by a set of descriptors that are based on frequency, physiochemical properties as well as topological and geometric substructures (fragments) [3, 4, 18, 25, 38, 42, 49].

Historically, the best performing and most widely used descriptors have been based on fingerprints, which represent each molecular graph by a fixed length bit-vector derived by enumerating all bounded length paths in the graph (e.g., Daylight [3]), fingerprints that consists of fragments of increasing size around atoms (Extended connectivity based descriptors [24,38]), and on sets of fragments that have been identified a priori by domain experts (e.g., Maccs keys [4, 14]). However, in recent years, research in the data mining community has generated new classes of descriptors based on frequently occurring substructures [18] and selected cycles & trees [25] that have been shown to achieve promising results.

In this paper, we try to understand which aspects of the molecular graph are important in providing effective descriptor-based representations in the context of SVM-based chemical compound classification and ranked-retrieval. We also study the effectiveness of various descriptor-based similarity measures for both deriving kernel functions for SVM-based classification and for ranked-retrieval. The five previously developed descriptors that we study are fingerprints [10], extended connectivity fingerprints [5, 38], Maccs keys [4], Cycles & trees [25] and frequent subgraph-based descriptors [18]. Each of these descriptors represent certain inherent choices that are made in designing any substructure based descriptor space. In order to better understand the strengths and weaknesses of the design choices and their impact on classification and retrieval performance, we also introduce a new set of fragment-based descriptors. These descriptors are derived from the set of all connected fragments present in the molecular graphs (graph fragments or GF) and three of its subsets.

We perform a detailed analysis of the design choices of these nine descriptors

and also conduct an experimental study on these descriptors using 28 different classification and retrieval problems derived from 18 datasets. Our study compares the performance achieved by the various descriptors and provides key insights on how the topology, discovery method, exactness and completeness of representation affects the performance. Our experiments also show that for both the classification and the retrieval tasks the GF descriptors are equivalent to extended connectivity fingerprints and consistently and statistically outperformed all the other methods studied in this paper. Moreover, a kernel function introduced in this paper that takes into account the length (size) of the fragments present in the set of descriptors lead to better overall results, especially when used with the GF-based descriptors.

The rest of the paper is organized as follows. Section 2 provides some background on the molecular graph representation of chemical compounds. Section 3 describes the previously developed descriptors. Section 4 provides a detailed description of the characteristics of various descriptor spaces. Section 5 describes the various descriptor spaces introduced in this paper. Section 6 provides a description of the various kernel functions used. Section 7 contains experimental evaluation techniques and results. Section 8 contains discussion of the various descriptors in the light of the results and provides concluding remarks on this work.

2. Representation of Compounds

In this paper we represent each compound by its corresponding molecular graph [29]. The vertices of these graphs correspond to the various atoms (e.g., carbon, nitrogen, oxygen, etc.), and the edges correspond to the bonds between the atoms (e.g., single, double, etc.). Each of the vertices and edges has a label associated with it. The labels on the vertices correspond to the type of atoms and the labels on the edges correspond to the type of bonds. Specifically, we use atomic numbers or a unique identifiers for each atomic number as the atom typing for vertices. For the edge labels, we use separate integers or identifiers for single, double and triple bonds. We also apply two commonly used structure normalization transformations [33]. First, we label all bonds in aromatic rings as *aromatic* (i.e., a different edge-label), and second, we remove the hydrogen atoms that are connected to carbon atoms (i.e., hydrogen-suppressed chemical graphs). To generate fingerprints and Maccs keys we use the Smiles [3] representation as an input.

3. Overview of Existing Descriptor Spaces

3.1. Fingerprints (fp-n)

Fingerprints are used to encode structural characteristics of a chemical compound into a fixed bit vector and are used extensively for various tasks in chemical informatics. These fingerprints are typically generated by enumerating all cycles and linear paths up to a given number of bonds and hashing each of these cycles and paths into a fixed bit-string [3, 10]. The specific bit-string that is generated depends on the number of bonds, the number of bits that are set, the hashing function, and the length of the bit-string. The key property of these

fingerprint descriptors is that they encode a very large number of sub-structures into a compact representation. Many variants of these fingerprints exist, some use predefined structural fragments in conjunction with the fingerprints (Unity fingerprints [6]), others count the number of times a bit position is set (hologram [7]), etc. In [30], it is shown that the performance of most of these fingerprints is comparable. We will refer to these descriptors as *fp-n* where n is the number of bits that are used.

3.2. Extended Connectivity Fingerprints (ECFP)

Molecular descriptors and fingerprints based on the extended connectivity concept have been described by several authors [24,38]. Recently, these fingerprints have been popularized by their implementation within Pipeline Pilot [5]. These fingerprints are generated by first assigning some initial label to each atom and then applying a Morgan type algorithm [35] to generate the fingerprints. Morgan's algorithm consists of l iterations. In each iteration, a new label is generated and assigned to each atom by combining the current labels of the neighboring atoms (i.e, connected via a bond). The union of the labels assigned to all the atoms over all the l iterations are used as the descriptors to represent each compound.

The key idea behind this descriptor generation algorithm is to capture the topology around each atom in the form of shells whose radius ranges from 1 to l. Thus, these descriptors can capture rather complex topologies. The value for l is a user supplied parameter and typically ranges from two to five.

To control the length of the labels they are often represented by fixed-width integers (e.g 32 bits), and the new label is generated by applying an arithmetic or logical operation on the labels of the neighboring atoms. As a result, the same label can potentially be assigned to multiple atoms, even when the topology of their surrounding shells are different. However, detailed studies have shown that such "collisions" are usually rare [5,38]. We will refer to this descriptor as *ECFP*.

3.3. Maccs Keys (MK)

Molecular Design Limited (MDL) created the key based fingerprints (Maccs Keys) [4] based on pattern matching of a chemical compound structure to a pre-defined set of structural fragments that have been identified by domain experts [19]. Each such structural fragment becomes a key and occupies a fixed position in the descriptor space. This approach relies on pre-defined rules to encapsulate the essential molecular descriptors a-priori and does not learn them from the chemical dataset. This descriptor space is notably different from fingerprint based descriptor space. Unlike fingerprints, no *folding* (hashing) is performed on the sub-structures. We will use the 166 structural keys by Molecular Design Limited (MDL) and will refer to this descriptor space as *MK*.

3.4. Cyclic patterns and Trees (CT)

Horovath *et al* [25] developed a method that is based on representing every compound as a set of cycles and certain kinds of trees. In particular, the idea is to

identify all the biconnected components (blocks) of a chemical graph. Once these blocks are identified, the first set of features is generated by enumerating up to a certain number of simple cycles (bounded cyclicity) for the blocks. Once the cycles are identified, all the blocks of the chemical graph are deleted. The resulting graph is a collection of leftover trees forming a forest. Each such tree is used as a descriptor. The final descriptor space is the union of the cycles and leftover trees. The tree patterns used in this representation are of a specific topology and size that depends on the position of blocks in the chemical graph. We will refer to this descriptor space as *CT*.

3.5. Frequent Sub-structures (FS)

A number of methods have been proposed in recent years to find frequently occurring sub-structures in a chemical graph database [26, 32, 36, 51]. Frequent sub-structures of a chemical graph database $\mathbf{D}$ are defined as all sub-structures that are present in at least σ ($\sigma \leq |\mathbf{D}|$) of compounds of the database, where σ is the absolute minimum frequency requirement (also called absolute minimum support constraint). These frequent sub-structures can be used as descriptors for the compounds in that database. One of the important properties of the sub-structures generated, like Maccs Keys, is that they can have arbitrary topology. Moreover, every sub-structure generated is connected and frequent (as determined by the minimum support constraint σ).

Descriptor space formed out of frequently occurring sub-structures depends on the value of σ. Therefore, unlike the Maccs keys, the descriptor space can change for a particular problem instance if the value of σ is changed. Moreover, unlike fingerprints, all frequent subgraphs irrespective of their size (number of bonds) form the descriptor space. A potential disadvantage of this method is that it is unclear how to select a suitable value of σ for a given problem. A very high value will fail to discover important sub-structures whereas a very low value will result in combinatorial explosion of frequent subgraphs. We will refer to this descriptor space as *FS*.

4. Characteristics of Descriptor Spaces

A careful analysis of the five descriptor spaces described in Section 3 illustrate four dimensions along which these schemes compare with each other and represent some of the choices that have been explored in designing fragment-based (or fragment-derived) descriptors for chemical compounds. Table 1 summarizes the characteristics of these descriptor spaces along the four dimensions. The first dimension is associated with whether the fragments are determined directly from the dataset at hand or they have been pre-identified by domain experts. Maccs keys is an example of a descriptor space whose fragments have been determined a priori whereas in all other schemes used in this study, the fragments are determined directly from the dataset. The advantage of an a priori approach is that sub-structures of arbitrary topology can form a part of the descriptor space. Moreover, the sub-structures selected encode domain knowledge in a compact descriptor space. But it also has a disadvantage of potentially not being able to adapt to the characteristics for a particular dataset and classification problem.

The second dimension is associated with the topological complexity of the ac-

tual fragments. On one end of the spectrum, schemes like fingerprints use rather simple topologies consisting of paths and cycles, whereas on the other end, frequent sub-structure-based descriptors allow topologies with arbitrary complexity. Topologically complex fragments along with simple ones might enrich the descriptor space.

The third dimension is associated with whether or not the fragments are being precisely represented in the descriptor space. For example, most schemes generate descriptors that are precise in the sense that there is a one-to-one mapping between the fragments and the dimensions of the descriptor space. In contrast, due to the hashing approach that they use or the fixed-length of their representation, descriptors such as fingerprints and extended connectivity fingerprints lead to imprecise representations (i.e., many fragments can map to the same dimension of the descriptor space). Depending on the number of these many-to-one mappings, these descriptors can lead to representations with varying degree of information loss.

Finally, the fourth dimension is associated with the ability of the descriptor space to cover all (or nearly all) of the dataset. Descriptor spaces created from fingerprints, extended connectivity fingerprints, and cycles & trees are guaranteed to contain fragments or hashed fragments from each one of the compounds. On the other hand, descriptor spaces corresponding to Maccs keys and frequent sub-structures may lead to a descriptor-based representation of the dataset in which some of the compounds have no (or a very small number) of descriptors. A descriptor space that covers all the compounds of a dataset has the advantage of encoding some amount of information for every compound.

From the above discussion it seems that descriptor spaces that are determined dynamically from the dataset, use fragments with simple and complex topologies, lead to precise representations, and have a high degree of coverage may be expected to perform better in the context of chemical compound classification and retrieval as they allow for a better representation of the underlying compounds. The descriptors that come closest to satisfying all the desirable properties are ECFP, CT and FS. ECFP virtually satisfies all of the properties except precise representation since there is the possibility of collisions [5, 38]. On the other hand, CT does not attempt to enumerate trees (only cycles are enumerated). Furthermore, the tree topologies depend on the location of blocks in the molecular graph. Lastly, FS suffers from potential incomplete coverage depending on the support threshold.

5. Graph Fragment based Descriptor Spaces

To better study the impact of the above design choices, we introduce a new descriptor space that we believe better captures the desired characteristics along the above four dimensions. Like FS, this descriptor space is determined dynamically from the dataset, the topology of the fragments that it consists of are arbitrary connected fragments and leads to a precise representation. However, unlike FS, which may suffer from partial coverage, the new descriptor is ensured to have 100% coverage by eliminating the minimum support criterion and generating all fragments. In order to control the exponential number of fragments generated we replace the minimum support criterion in FS with an upper bound. Thus, this descriptor space consists of all connected fragments up to a given length l (i.e., number of bonds) that exist in the dataset at hand. We will refer to this

Table 1. Design choices made by the descriptor spaces.

Previously developed descriptors				
	Topological Complexity	Generation	Precise	Complete Coverage
MK	Low to High	static	Yes	Maybe
fp	Low	dynamic	No	Yes
CT	Medium	dynamic	Yes	Yes
FS	Low to High	dynamic	Yes	Maybe
ECFP	Low to High	dynamic	Maybe	Yes

GF-based descriptors				
	Topological Complexity	Generation	Precise	Complete Coverage
PF	Low	dynamic	Yes	Yes
TF	Medium	dynamic	Yes	Maybe
AF	Medium	dynamic	Yes	Yes
GF	Low to High	dynamic	Yes	Yes

descriptor space as *Graph Fragments (GF)*. The algorithm to efficiently generate this descriptor space is described in Appendix A.

In addition, we also derive three other sets of fragments from the set of all graph fragments. The first, termed as *Tree Fragments* (TF), is the collection of all fragments that have at least one node of degree greater than two and contains no cycles. This set forms all the tree fragments. The second set, called *Path Fragments* (PF), is just the set of linear paths where the degree of every node in every fragment is less than or equal to two. The third set of fragments, called *Acyclic Fragments* (AF) are derived such that AF = TF $\cup$ PF. Table 1 also provides a description of their properties in terms of design choices. It should be pointed out that TF descriptors may lead to incomplete coverage when a compound is itself a linear path of atoms.

Note that Path Fragments are exactly the same patterns as the linear paths in fingerprints [10] and the path-based generalized fingerprints in [42]. But [10] and [42] also use cycles along with the linear paths. Also note that acyclic fragments (AF) are also referred to as free trees. Another important observation is that any frequent sub-structure based descriptor space is a superset of Graph-Fragments when the minimum support threshold (σ) is one.

6. Descriptor-based Kernel Functions

Given the descriptor space, each chemical compound can be represented by a vector X whose i^{th} dimension will have a non-zero value if the compound contains that descriptor and will have a value of zero otherwise. The value for each descriptor that is present can be either one, leading to a vector representation that captures presence or absence of the various descriptors (referred to as binary vectors) or the number of times (number of embeddings) that each descriptor occurs in the compound, leading to a representation that also captures the frequency information (referred to as frequency vectors).

Given the above vector representation of the chemical compounds, the classification algorithms that we develop in this paper use support vector machines (SVM) [44] as the underlying learning methodology, as they have been shown to be highly effective, especially in high dimensional spaces. One of the key parameters that affects the performance of SVM is the choice of the kernel function ($\mathcal{K}$), that measures the similarity between pairs of compounds. Any function can

be used as a kernel as long as, for any number n and any possible set of distinct compounds $\{X_1, \ldots, X_n\}$, the $n \times n$ Gram matrix defined by $\mathcal{K}_{i,j} = \mathcal{K}(X_i, X_j)$ is symmetric positive semidefinite. These functions are said to satisfy Mercer's conditions and are called Mercer kernels, or simply valid kernels.

In this paper we use the Min-Max kernel [42] as our choice of the kernel function. This kernel was selected because it has been shown to be an effective way to measure the similarity between chemical compound pairs and outperform Tanimoto coefficient [42] (which is the most widely used kernel function in cheminformatics) in empirical evaluations. Given the vector representation of two compounds X and Y, the Min-Max kernel function is given by

$$\mathcal{K}_{MM}(X,Y) = \frac{\sum_{i=1}^{M} min(x_i, y_i)}{\sum_{i=1}^{M} max(x_i, y_i)}, \tag{1}$$

where the terms x_i and y_i are the values along the i^{th} dimension of the X and Y vectors, respectively. Note that in the case of binary vectors, these will be either zero or one, whereas in the case of frequency vectors these will be equal to the number of times the i^{th} descriptor exists in the two compounds. Moreover, note that the Min-Max kernel is a valid kernel as it has been shown to satisfy Mercer's conditions [42] and reduces to Tanimoto kernel in the case of binary vectors.

One of the potential problems in using the above kernel with descriptor spaces that contain fragments of different lengths is that they contain no mechanism to ensure that descriptors of various lengths contribute in a non-trivial way to the computed kernel function values. This is especially true for the GF descriptor space and its subsets in which each compound tends to have a much larger number of longer length fragments (e.g. length six and seven) than shorter length (e.g. length two and three). To overcome this problem we modified the above kernel function to give equal weight to the fragments of each length. Particularly, for the Min-Max kernel function, this is obtained as follows. Let X^l and Y^l be the feature vectors of X and Y with respect to only the features of length l, and let L be the length of the largest feature. Then, the length-differentiated Min-Max kernel function $\mathcal{K}^*_{MM}(X,Y)$ is given by

$$\mathcal{K}^*_{MM}(X,Y) = \frac{1}{L}\sum_{l=1}^{L} \mathcal{K}_{MM}(X^l, Y^l). \tag{2}$$

We will refer to this as the *length-differentiated kernel function*, and we will refer to the one that do not differentiate between different length fragments as *pooled kernel function*.

In summary, we studied four different flavors for the Min-Max kernel function, one that is binary and pooled, frequency and pooled, binary and length-differentiated and frequency and length-differentiated. We also studied these four flavors of RBF kernel, but the results were worse than Min-Max [46] so we are not including them here. We will follow the convention of using the symbols $\mathcal{K}_b$, $\mathcal{K}_f$, $\mathcal{K}^*_b$, and $\mathcal{K}^*_f$ to refer to binary and pooled, frequency and pooled, binary and length-differentiated and frequency and length-differentiated Min-Max kernel functions, respectively.

Table 2. Properties of classification problems and Datasets.

D	N	$N+$	N_A	N_{A+}	N_{A-}	N_B	N_{B+}	N_{B-}
NCI1	38311	1805	26	34	25	28	37	27
NCI109	37085	1613	26	34	25	28	37	27
NCI123	36477	2552	26	32	25	28	34	27
NCI145	36594	1512	26	34	25	28	37	27
NCI167	73464	8648	21	24	21	22	25	22
NCI220	723	232	24	24	25	26	25	26
NCI33	36617	1239	26	35	25	28	38	27
NCI330	37877	1913	22	28	21	23	30	23
NCI41	25049	1165	26	35	26	28	38	28
NCI47	36857	1561	26	34	25	28	37	27
NCI81	37124	1881	26	33	25	28	36	27
NCI83	25240	1805	26	33	25	28	35	28
H1	37913	1157	27	37	26	29	39	28
H2	37061	294	27	43	26	29	45	28
A1	34827	12374	25	25	25	25	25	25
H3	1158	293	37	43	34	39	45	37
D1	988	82	24	27	23	25	28	25
D2	990	84	24	25	23	25	27	25
D3	1101	191	26	36	23	28	38	25
D4	1264	360	26	32	23	28	34	25
P1	407	138	18	17	19	19	18	20
P2	415	110	19	17	19	19	18	20
P3	393	103	18	16	19	19	17	20
P4	410	119	18	17	19	19	17	20
C1	604	301	14	13	15	14	14	15
M1	1458	268	16	14	16	16	15	17
M2	1458	163	16	13	16	16	14	17
M3	1458	85	16	13	16	16	13	17

N is the total number of compounds in the dataset. $N+$ is the number of positives in the dataset. N_A and N_B are the average number of atoms and bonds in each compound. N_{A+} is the average number of atoms in each compound belonging to the positive class and N_{A-} is the average number of atoms in each compound belonging to the negative class. Similarly N_{B+} and N_{B-} are the corresponding numbers for bonds. The numbers are rounded off to the nearest integer.

7. Results

7.1. Datasets

The performance of the different descriptors and kernel functions was assessed on 28 different classification problems from 18 different datasets. The size, distribution and compound characteristics of the 28 classification problems are shown in Table 2. Each of the 28 classification problems is unique in that it has different distribution of positive class (ranging from 1% in H2 to 50% in C1), different number of compounds (ranging from the smallest with 559 compounds to largest with 78,995 compounds) and compounds of different average sizes (ranging from the 14 atoms per compound to 37 atoms per compound on an average in C1 and H3 respectively).

The first data set that was used is a part of the Predictive Toxicology Evaluation Challenge [11, 40]. It contains data published by the US National Institute for Environmental Health Sciences and consists of bio-assays of different chemical compounds on rodents to study the carcinogenicity properties of the compounds. Each compound is evaluated on male rats, female rats, male mice, and female mice, and is assigned class labels indicating the toxicity or non-toxicity of the

Table 3. Description of NCI cancer screen datasets.

Name (Bioassay-ID or AID)	Description
NCI-H23 (NCI1)	Human tumor (Non-Small Cell Lung) cell line growth inhibition assay
OVCAR-8 (NCI109)	Human tumor (Ovarian) cell line growth inhibition assay
MOLT-4 (NCI123)	Human tumor (Leukemia) cell line growth inhibition assay
SN12C (NCI145)	SN12C Renal cell line
Yeast anti-cancer (NCI167)	Yeast anti-cancer screen bub3 strain
CD8F1 (NCI220)	In Vivo Anticancer Screen Tumor model Mammary Adenocarcinoma
UACC257 (NCI33)	Human tumor (Melanoma) cell line growth inhibition assay
P388 in CD2F1 (NCI330)	In Vivo Anticancer Screen tumor model P388 Leukemia (intraperitoneal)
PC-3 (NCI41)	Human tumor (Prostate) cell line growth inhibition assay
SF-295 (NCI47)	Human tumor (Central Nervous System) cell line growth inhibition assay
SW-620 (NCI81)	Human tumor (Colon) cell line growth inhibition assay
MCF-7 (NCI83)	Human tumor (Breast) cell line growth inhibition assay

compound for that animal. We derive four problem sets out of this dataset, one corresponding to each of the rodents mentioned above. These will be referred to as *P1*, *P2*, *P3*, and *P4*.

The second dataset used in this paper is mutagenicity data from [11,22]. The mutagenicity data set was extracted from the carcinogenic potency database (CPDB) [20] and provides mutagenicity classes (mutagens and nonmutagens) as determined by the Salmonella/microsome assay (Ames test [13]). The problem for this dataset is to distinguish between these two classes. We will refer this dataset as *C1*.

The third data set is obtained from the National Cancer Institutes DTP AIDS Antiviral Screen program [1, 31]. Each compound in the data set is evaluated for evidence of anti-HIV activity. Compounds that provided at least 50 percent protection were listed as confirmed moderately active (CM). Compounds that reproducibly provided 100 percent protection were listed as confirmed active (CA). Compounds neither active nor moderately active were listed as confirmed inactive (CI). We formulated three problems out of this dataset. The first problem is designed to distinguish between CM+CA and CI; the second between CA and CI, and the third between CA and CM. We will refer to these problems as *H1*, *H2*, and *H3*, respectively.

The fourth data set was obtained from the Center of Computational Drug Discoverys anthrax project at the University of Oxford [37]. The goal of this project was to discover small molecules that would bind with the heptameric protective antigen component of the anthrax toxin, and prevent it from spreading its toxic effects. The screen identified a set of 12,376 compounds that could potentially bind to the anthrax toxin and a set of 22,460 compounds that were unlikely to bind to the toxin. The task for this data set was to identify if a given a chemical compound will bind the anthrax toxin (active) or not (inactive). This dataset problem is referred as *A1*.

A fifth dataset used in this paper consists of 1728 currently marketed drugs and each drug compound in this dataset is marked either as Oral (O), Topical (T), Absorbent (A) or Injectable (I) depending on the mode of administration of

that drug. This dataset was compiled mostly from the FDA's Orange book [12] and the MDL database [8]. A detailed description of this dataset can be found in [45]. Four tasks are defined from this dataset: to distinguish between Oral and Absorbent *D1*, between Oral and Topical *D2*, between Oral and Injectable *D3* and between Oral and everything else (Topical + Absorbent + Injectable) as *D3*.

Another dataset used in this study is the MAO (Monoamine Oxidase) dataset [17]. Monoamine Oxidase are enzymes that catalyze the oxidation of neurotransmitters and neuromodulator called monoamines. This dataset consists of compounds that are Mono amine Oxidase inhibitors. The compounds of this dataset have been categorized into four different classes (0, 1, 2 and 3) based on the levels of activity, with the lowest labeled as 0 (inactive) and the highest labeled as 3 (highest potency), all based on the IC50 values of each compound in a MAO assay. We derive three problems from this dataset: *M1* with positive class compounds as labels 1, 2 and 3 and negative class as compounds with label 0, *M2* with positive class as labels 2 and 3 and negative class compounds as labels 0 and 1, and finally the last problem *M3* with positive class compounds as label 3 and rest of the compounds in negative class.

The rest of the datasets are derived from the PubChem website that pertain to cancer cell lines [9]. Twelve datasets are selected from the bioassay records for twelve different types of cancer cell lines. Each of the NCI anti-cancer screens forms a classification problem. Since there is more than one screen available for any particular type of cancer (for example colon cancer, breast cancer *etc.*), we decided to use the screen that had the most number of compounds tested on it. Each of these bioassay records have information on the assay type, compound identifier, activity score, outcome etc. as submitted by the depositor of the bioassay screen. The class labels on these datasets is decided by the "outcome" field of the bioassay which is either active or inactive. We used the original class labels associated with each compound for this study. Table 3 proves details of the 12 different bioassays used for this study.

All the datasets required some data cleaning. For some of the compounds we were unable to generate the fingerprints due to the use of third party software for fingerprint generation and impossible kekule forms and serious valence errors in raw data. Furthermore, many compounds in these datasets were non drug-like, in that, they contained elements such as arsenic, lead etc. All such compounds were removed from their respective datasets. The dataset cleaning made the sets of compounds used for different descriptors exactly the same and allowed objective comparison of the descriptor spaces. Another important observation is that the active compounds in almost all the datasets used in this study do not fall into a particular target activity class. In most assays, the target is either unknown (for example NCI cancer and HIV assays, anthrax and toxicity datasets etc.) or the classification and retrieval problems are defined to be target non-specific (for example Drug dataset). The only exception to this is the MAO dataset that consists of Monoamine Oxidase inhibitors.

7.2. Experimental Methodology

7.2.1. Classification Task

The classification results were obtained by performing a 5-way cross validation on the dataset, ensuring that the class distribution in each fold is identical to the original dataset. In each one of the cross validation experiments, the test-set was never considered and the algorithm used only the training-set to generate the descriptor space representation and to build the classification model. The exact same training and test sets were used in descriptor generation and cross validation experiments for all the different schemes. The SVM classifier experiments were run on Dual Core AMD Opterons with 4 GB of memory. For the SVM classifier we used the SVMLight library [27] with all the default parameter settings except the kernel.

7.2.2. Retrieval Task

We also compare the effectiveness of the different descriptor spaces for the task that is commonly referred to as a ranked-retrieval or database screening [48]. The goal of this task is, given a compound that has been experimentally determined to be active, to find other compounds from a database that are active as well. Since the activity of a chemical compound depends on its molecular structure, and compounds with similar molecular structure tend to have similar chemical function, this task essentially maps to ranking the compounds in the database based on how similar they are to the *query* compound. In our experiments, for each dataset we used each of its active compounds as a query and evaluated the extent to which the various descriptor spaces along with the kernel functions studied in this paper lead to similarity measures that can successfully retrieve the other active compounds. Notice that all the kernel functions described in Section 6 are valid similarity measures.

7.2.3. Descriptor Generation

All descriptors were generated on a Pentium 2.6 GHz machine with 1 GB memory. For fingerprints, we used Chemaxon's fingerprint program called Screen [10]. We experimented using 256-, 512-, 1024-, 2048-, 4196- and 8192-bit length fingerprints. We used default settings of the two parameters: number of bonds or maximum length of the pattern generated (up to seven) and number of bits set by a pattern (three). We found that 8192-bits produced better results (even though their performance advantage was not statistically significant compared to 2048- and 4196-bit fingerprints). For this reason, we use 8192-bit fingerprints (fp-n where $n = 8192$) in all the comparisons against other descriptors.

ECFP's were generated using a multiplicative form of Morgan's algorithm. The type of descriptor space generated by this algorithm is calibrated by two variables: (i) the initial atom label used to describe each atom and (ii) the maximum shell radius (i.e, the farthest atom considered in terms of bond distance). For our study we used the atomic number as the initial label for each atom and a maximum shell radius of three. Thus the fragments that form the ECFP descriptor space are a union of all the fragments formed by taking all atoms one, two and three bond distance away from every atom.

To generate MDL Maccs keys (166 keys) we use the MOE suite by Chemical

Table 4. Support values for FS.

Datasets	$\sigma_-\%$	$\sigma_+\%$	*Datasets*	$\sigma_-\%$	$\sigma_+\%$
NCI1	5.0	7.0	A1	5.0	3.0
NCI109	4.0	4.0	H3	8.0	8.0
NCI123	4.0	5.0	D1	5.0	10.0
NCI145	4.0	6.0	D2	5.0	32.0
NCI167	2.0	2.0	D3	5.0	10.0
NCI220	5.0	8.0	D4	5.0	12.0
NCI33	4.0	4.0	P1	3.0	3.0
NCI330	4.0	8.0	P2	3.0	3.0
NCI41	4.0	6.0	P3	3.0	3.0
NCI47	4.0	5.0	P4	3.0	3.0
NCI81	5.0	6.0	C1	2.0	2.0
NCI83	4.0	4.0	M1	1.5	1.75
H1	8.0	5.0	M2	1.45	1.5
H2	8.0	8.0	M3	1.25	3.0

Computing Group [2]. For Cyclic patterns and Trees, we use 1000 as the upper bound on the number of cycles to be enumerated as described in [25] in our own implementation of the algorithm. To generate frequent sub-structures, we use the FSG algorithm described in [32], although any other frequent subgraph discovery algorithm could be used. Table 4 contains the values of σ used for positive and negative classes in each dataset. Most of the support values are the same or lower than in [18] for the common datasets and are derived in the same fashion as described in [18]. The lowest support value was selected that could allow FSG to use a reasonable amount of time and memory.

7.2.4. Kernel Functions

In the context of fp-n the only kernel applicable is the binary and pooled ($\mathcal{K}_b$) kernel. This is because these hashed fingerprints are inherently binary and do not provide frequency information. In the context of ECFP and MK, only two kernels ($\mathcal{K}_b$ and $\mathcal{K}_f$) are applied as the length information for ECFP and Maccs keys were not available. For the rest of the descriptor spaces (GF, AF, TF, PF, CT and FSG), we applied all the four kernels described in Section 6.

7.3. Performance Assessment Measures

The classification performance was assessed by computing the ROC50 values [21], which is the area under the ROC curve up to the first 50 false positives. This is a much more appropriate performance assessment measure than traditional ROC value for datasets with very small positive classes. This is because for such problem settings, a user will most likely stop examining the highest scoring predictions as soon as he/she starts encountering a certain number of false positives [21].

We assess the ability of a particular descriptor set to identify positive compounds in the context of ranked-retrieval task by looking at the fraction of positive compounds that were recovered in the top k retrieved compounds. Specifically, we report the fraction of positives recovered in the top k retrieved compounds in a ranked-retrieval task in which every positive compound is used as query. We call this metric *normalized hit rate* (NHR) and it is computed as follows. Suppose N is the number of compounds in a dataset, N_+ is the number of positive (active) compounds in that dataset and $hits_k$ is the number of positives

found in the top k retrieved compounds over all queries. Then, the normalized hit rate is given by

$$\text{NHR} = \frac{hits_k}{(kN_+)}. \tag{3}$$

To compare the performance of a set of schemes across the different datasets, we compute a summary statistic that we refer to as the *Average Relative Quality to the Best (ARQB)* as follows: Let $r_{i,j}$ be the ROC50 (NHR) value achieved by the scheme j on the dataset i, and let r_i^* be the maximum (i.e. the best) ROC50 (NHR) value achieved for this dataset over all the schemes. Then the ARQB for scheme j is equal to $(1/T)\left(\sum_i (r_{i,j}/r_i^*)\right)$, where T is the number of datasets. An ARQB value of one indicates that the scheme achieved the best results for all the datasets compared to the other schemes, and a low ARQB value indicates a poorly performing scheme.

We used the Wilcoxon's paired signed-rank test [15] to compare the statistical significance of any two descriptors based on the performance measures described above. A p-value of 0.01 is used as threshold for all comparisons.

7.4. Evaluation of GF Descriptors

7.4.1. Complexity of GF Descriptors Generation

Table 5 shows the number of graph fragments (GF) of various lengths that were generated for each dataset as well as the time required to generate the fragments of length seven. These results show that the number of fragments does increase considerably with l, which essentially puts a practical upper bound on the length of the fragments that can be used for classification. In fact, for $l = 8$ (not shown here), the number of fragments were about three to five times more than that for $l = 7$, which made it impractical to build SVM-based classifier for many of the datasets. However, on the positive side, the amount of time required to generate these fragments is reasonable, and is significantly lower than that required for learning the SVM models.

7.4.2. Sensitivity on the Length of GF Descriptors

To evaluate the impact of the fragment length on the performance achieved by the GF descriptors for classification and retrieval, we performed a study in which we varied the maximum fragment length l from two to seven bonds. The results of this study are shown in Table 6 and 7. These results were obtained using the $\mathcal{K}_f^*$ kernel, which as will be shown later, is one of the best performing kernels for GF descriptors.

From the results in Table 6 we can see that the classification performance tends to improve as l increases, and the scheme that use up to length seven fragments achieve the best overall performance (in terms of ARQB). Moreover, all of these differences are statistically significant. On the other hand the retrieval performance in terms of ARQB, as shown in Table 7, saturates as l increases from six to seven. Also, results with l equal to five, six and seven are not statistically different from each other for ranked-retrieval. This indicates that, for ranked-retrieval task, larger fragments do not improve performance in the context of GF fragments.

Table 5. Numbers of GF for different lengths l.

D	# of fragments $l = 3$	$l = 5$	$l = 7$	runtime (in sec) for $l = 7$
NCI1	6277	97040	1068091	1181
NCI109	6305	97322	1069998	1183
NCI123	6196	95886	1055260	1173
NCI145	6277	96600	1061262	1173
NCI167	8564	12472	1292111	1567
NCI220	1575	13272	86000	27
NCI33	6222	96280	1060702	1180
NCI330	7400	10252	987743	891
NCI41	5329	81112	862559	830
NCI47	6255	96725	1064385	1190
NCI81	6297	97095	1070018	1201
NCI83	5367	81632	867034	823
H1	14387	17143	1420543	1529
H2	14266	16968	1402533	1494
A1	3233	66639	733125	504
H3	2760	23781	140901	74
D1	2129	19068	105886	31
D2	2120	18719	103495	32
D3	2246	20780	120560	42
D4	2339	21849	127194	48
P1	1220	8017	37973	10
P2	1241	8146	38742	10
P3	1242	8053	38051	10
P4	1242	8009	37586	9
C1	1137	6575	30081	7
M1	1306	9661	40186	11
M2	1306	9661	40186	11
M3	1306	9661	40186	11

We have omitted the results for l equal to 2, 4 and 6 as they fit into a similar trend.

7.4.3. Effectiveness of Different Kernels for GF Descriptor

Table 8 and 9 shows the classification and ranked-retrieval performance of the different kernel functions described in Section 6 for the GF descriptors. These results were obtained for GF descriptors containing fragments of length up to seven.

These results show that the best performing kernel function is the $\mathcal{K}_f^*$ (length-differentiated frequency vectors). Thus, giving equal weights to the fragments of various lengths leads to better results. Note that for classification results in Table 8, based on the Wilcoxon statistical test of $p = 0.01$, the differences between $\mathcal{K}_b^*$ and $\mathcal{K}_f^*$ are not significant, but $\mathcal{K}_f^*$ is statistically better than $\mathcal{K}_b$. Also, $\mathcal{K}_f^*$ is statistically better than $\mathcal{K}_f$ at $p = 0.05$. Moreover, the table shows that including frequency information leads to better results. For ranked-retrieval results in Table 9, $\mathcal{K}_f^*$ is statistically better than $\mathcal{K}_b$ at $p = 0.05$. But all the other pairwise comparisons of the different kernels show statistically equivalent performance to each other.

Table 6. ROC50 results for the $\mathcal{K}_f^*$ kernel for different lengths using GF descriptors.

D	up to $l=2$	up to $l=3$	up to $l=4$	up to $l=5$	up to $l=6$	up to $l=7$
NCI1	0.294	0.298	0.306	0.313	0.321	0.329
NCI109	0.255	0.277	0.292	0.305	0.312	0.317
NCI123	0.230	0.245	0.247	0.253	0.262	0.269
NCI145	0.302	0.316	0.332	0.346	0.358	0.369
NCI167	0.047	0.056	0.058	0.061	0.064	0.065
NCI220	0.295	0.288	0.286	0.291	0.288	0.287
NCI33	0.254	0.274	0.293	0.310	0.320	0.328
NCI330	0.311	0.339	0.351	0.358	0.361	0.363
NCI41	0.286	0.312	0.329	0.339	0.348	0.358
NCI47	0.249	0.270	0.283	0.296	0.307	0.314
NCI81	0.237	0.256	0.265	0.270	0.272	0.276
NCI83	0.255	0.280	0.293	0.300	0.302	0.311
H1	0.236	0.246	0.250	0.253	0.255	0.259
H2	0.561	0.572	0.584	0.592	0.601	0.609
A1	0.138	0.138	0.154	0.170	0.200	0.207
H3	0.630	0.638	0.646	0.651	0.657	0.660
D1	0.252	0.247	0.239	0.253	0.263	0.278
D2	0.575	0.584	0.595	0.600	0.605	0.601
D3	0.485	0.489	0.493	0.492	0.498	0.501
D4	0.446	0.463	0.477	0.481	0.484	0.487
P1	0.676	0.687	0.694	0.693	0.687	0.686
P2	0.634	0.646	0.651	0.651	0.656	0.659
P3	0.632	0.643	0.642	0.644	0.643	0.648
P4	0.654	0.660	0.666	0.668	0.667	0.669
C1	0.780	0.789	0.802	0.811	0.823	0.830
M1	0.484	0.490	0.487	0.476	0.472	0.472
M2	0.635	0.654	0.658	0.659	0.659	0.663
M3	0.777	0.784	0.784	0.781	0.784	0.787
ARQB	0.891	0.925	0.946	0.965	0.983	0.997

Table 7. NHR ($k = 50$) results for the $\mathcal{K}_f^*$ kernel for different lengths using GF descriptors.

D	up to $l=2$	up to $l=3$	up to $l=4$	up to $l=5$	up to $l=6$	up to $l=7$
NCI1	0.313	0.327	0.335	0.340	0.341	0.342
NCI109	0.280	0.296	0.306	0.309	0.309	0.310
NCI123	0.295	0.308	0.314	0.315	0.315	0.315
NCI145	0.295	0.314	0.324	0.327	0.328	0.329
NCI167	0.216	0.221	0.223	0.224	0.225	0.225
NCI220	0.303	0.305	0.303	0.303	0.302	0.304
NCI33	0.243	0.252	0.257	0.259	0.259	0.260
NCI330	0.334	0.345	0.351	0.356	0.360	0.363
NCI41	0.263	0.279	0.289	0.293	0.295	0.296
NCI47	0.291	0.307	0.316	0.319	0.320	0.321
NCI81	0.288	0.304	0.312	0.314	0.315	0.314
NCI83	0.285	0.297	0.304	0.309	0.312	0.313
H1	0.201	0.205	0.209	0.213	0.214	0.215
H2	0.265	0.278	0.290	0.296	0.299	0.298
A1	0.568	0.580	0.588	0.592	0.593	0.593
H3	0.524	0.533	0.535	0.540	0.541	0.541
D1	0.109	0.110	0.113	0.115	0.118	0.119
D2	0.243	0.249	0.251	0.250	0.252	0.251
D3	0.224	0.228	0.232	0.236	0.237	0.237
D4	0.355	0.360	0.366	0.369	0.370	0.371
P1	0.381	0.387	0.390	0.391	0.389	0.389
P2	0.297	0.301	0.303	0.302	0.302	0.300
P3	0.312	0.313	0.314	0.317	0.315	0.313
P4	0.318	0.326	0.328	0.327	0.327	0.327
C1	0.517	0.519	0.520	0.522	0.525	0.531
M1	0.333	0.332	0.329	0.335	0.331	0.327
M2	0.312	0.326	0.319	0.310	0.319	0.310
M3	0.339	0.337	0.340	0.340	0.337	0.337
ARQB	0.944	0.971	0.985	0.992	0.996	0.996

Table 8. ROC50 values for the GF descriptors using the different kernel functions.

Datasets	$(\mathcal{K}_b)$	$(\mathcal{K}_f)$	$(\mathcal{K}_b^*)$	$(\mathcal{K}_f^*)$
NCI1	0.326	0.326	0.321	0.329
NCI109	0.313	0.310	0.305	0.317
NCI123	0.255	0.263	0.258	0.269
NCI145	0.362	0.365	0.363	0.369
NCI167	0.059	0.061	0.060	0.065
NCI220	0.259	0.279	0.286	0.287
NCI33	0.317	0.335	0.313	0.328
NCI330	0.352	0.353	0.356	0.363
NCI41	0.356	0.372	0.348	0.358
NCI47	0.308	0.322	0.306	0.314
NCI81	0.267	0.274	0.266	0.276
NCI83	0.293	0.308	0.297	0.311
H1	0.247	0.258	0.249	0.259
H2	0.607	0.616	0.602	0.609
A1	0.194	0.202	0.198	0.207
H3	0.670	0.666	0.659	0.660
D1	0.330	0.293	0.299	0.278
D2	0.577	0.580	0.591	0.601
D3	0.472	0.486	0.482	0.501
D4	0.492	0.485	0.497	0.487
P1	0.698	0.681	0.703	0.686
P2	0.644	0.662	0.645	0.659
P3	0.671	0.648	0.675	0.648
P4	0.651	0.656	0.656	0.669
C1	0.811	0.823	0.815	0.830
M1	0.458	0.465	0.469	0.472
M2	0.634	0.641	0.654	0.663
M3	0.791	0.785	0.788	0.787
ARQB	0.966	0.978	0.971	0.987

Table 9. NHR ($k = 50$) values for the GF descriptors using the different kernel functions.

Datasets	$(\mathcal{K}_b)$	$(\mathcal{K}_f)$	$(\mathcal{K}_b^*)$	$(\mathcal{K}_f^*)$
NCI1	0.334	0.333	0.333	0.342
NCI109	0.304	0.306	0.305	0.310
NCI123	0.306	0.309	0.311	0.315
NCI145	0.323	0.324	0.323	0.329
NCI167	0.222	0.214	0.215	0.225
NCI220	0.314	0.315	0.314	0.304
NCI33	0.252	0.251	0.251	0.260
NCI330	0.358	0.349	0.346	0.363
NCI41	0.291	0.290	0.289	0.296
NCI47	0.314	0.313	0.310	0.321
NCI81	0.306	0.307	0.307	0.314
NCI83	0.308	0.308	0.309	0.313
H1	0.217	0.226	0.222	0.215
H2	0.292	0.304	0.309	0.298
A1	0.592	0.584	0.586	0.593
H3	0.546	0.536	0.538	0.541
D1	0.117	0.124	0.130	0.119
D2	0.250	0.237	0.243	0.251
D3	0.242	0.251	0.250	0.237
D4	0.376	0.374	0.372	0.371
P1	0.385	0.384	0.383	0.389
P2	0.286	0.288	0.293	0.300
P3	0.311	0.302	0.302	0.313
P4	0.326	0.327	0.330	0.327
C1	0.540	0.531	0.532	0.531
M1	0.329	0.331	0.328	0.327
M2	0.310	0.323	0.326	0.310
M3	0.336	0.368	0.380	0.337
ARQB	0.974	0.979	0.982	0.983

7.5. Comparison of Descriptor Spaces

7.5.1. Classification Performance

To compare the classification performance of the various descriptor spaces we performed a series of experiments in which all the kernels described in Section 6 that can be used in conjunction with the nine descriptor spaces (fp-n, MK, CT, FS, ECFP, GF, AF, TF, and PF) are employed to classify the various datasets. In order to objectively compare these nine schemes, in Table 10 we only compare the ROC50 results achieved by the two kernels (binary and pooled $\mathcal{K}_b$ and frequency and pooled $\mathcal{K}_f$) that are applicable to most of the descriptor spaces. In addition, Table 11 shows whether or not these schemes in combination with the kernels used achieve ROC50 results that are statistically different from each other. The results for GF, AF, TF, and PF were obtained for fragments up to length seven.

Comparing between the different descriptors, these results show that the GF, AF and ECFP descriptors lead to ROC50 results that are statistically better than that achieved by all other previously developed schemes. From a statistical point, GF, AF and ECFP descriptors is equivalent to each other for the same kernel function. In addition, the performance achieved by both TF and PF is also good and in general better than that achieved by the earlier approaches.

Comparing between fp-n, CT, MK, and FS, we can see that the fingerprint descriptors achieve the best overall results, whereas MK tend to perform the worst. However, from a statistical significance standpoint CT, MK, and FS are equivalent.

Another interesting observation is that the PF scheme ($\mathcal{K}_b$) achieves better results than fp-n (the difference is significant at $p = 0.05$). Since the fp-n descriptors were also generated by enumerating paths of length up to seven (and also cycles), the performance difference suggests that the folding that takes place due to the fingerprint's hashing approach negatively impacts the classification performance. This result is in agreement with [42] albeit we perform this comparison on a much higher number of datasets.

Comparing the performance between the two kernels ($\mathcal{K}_b$ and $\mathcal{K}_f$) for different descriptors it can be observed that $\mathcal{K}_f$ generally performs better than $\mathcal{K}_b$ in terms of ARQB. The differences between the two kernels is generally statistically significant at $p = 0.05$ for most descriptors, although it is not significant $p = 0.01$. Thus, adding frequency information helps in improving classification performance.

7.5.2. Retrieval Performance

For the task of ranked-retrieval we experimented using all possible combinations of the nine descriptor spaces and four kernel functions. Again, due to the reason mentioned in the section Section 7.5.1, in Table 12 we only show the NHR results for ($\mathcal{K}_b$ and $\mathcal{K}_f$) for each descriptor space. Table 13 shows the extent to which the relative performance of various schemes are statistically significant.

Comparing these results with those for the classification task shows similar trends with respect to the relative performance of the various descriptor spaces. In the case of ranked-retrieval, the ECFP descriptor with $\mathcal{K}_f$ is the best scheme in terms of ARQB outperforming most of the other schemes. Also, GF-based descriptors (GF, AF and TF) and ECFP are statistically equivalent and out-

Table 10. ROC50 values for the nine descriptors using $\mathcal{K}_b$ and $\mathcal{K}_f$ kernels.

Datasets	GF ($\mathcal{K}_b$)	GF ($\mathcal{K}_f$)	AF ($\mathcal{K}_b$)	AF ($\mathcal{K}_f$)	TF ($\mathcal{K}_b$)	TF ($\mathcal{K}_f$)	PF ($\mathcal{K}_b$)	PF ($\mathcal{K}_f$)	fp-n ($\mathcal{K}_b$)	ECFP ($\mathcal{K}_b$)	ECFP ($\mathcal{K}_f$)	CT ($\mathcal{K}_b$)	CT ($\mathcal{K}_f$)	MK ($\mathcal{K}_b$)	MK ($\mathcal{K}_f$)	FS ($\mathcal{K}_b$)	FS ($\mathcal{K}_f$)
NCI1	0.302	0.323	0.302	0.323	0.309	0.320	0.292	0.310	0.294	0.318	0.322	0.273	0.288	0.184	0.202	0.266	0.266
NCI109	0.284	0.309	0.284	0.309	0.291	0.305	0.263	0.294	0.272	0.304	0.315	0.240	0.237	0.210	0.217	0.251	0.259
NCI123	0.254	0.262	0.254	0.262	0.245	0.252	0.232	0.261	0.246	0.260	0.269	0.224	0.243	0.209	0.203	0.228	0.230
NCI145	0.336	0.351	0.334	0.351	0.338	0.347	0.313	0.335	0.304	0.349	0.353	0.285	0.283	0.217	0.224	0.289	0.295
NCI167	0.060	0.060	0.058	0.060	0.059	0.060	0.056	0.058	0.061	0.057	0.055	0.043	0.041	0.048	0.051	0.060	0.061
NCI220	0.302	0.274	0.304	0.275	0.285	0.289	0.290	0.282	0.332	0.312	0.276	0.256	0.261	0.504	0.517	0.218	0.213
NCI33	0.278	0.301	0.275	0.301	0.280	0.304	0.269	0.294	0.257	0.290	0.305	0.245	0.255	0.208	0.214	0.244	0.252
NCI330	0.349	0.356	0.336	0.355	0.327	0.344	0.325	0.348	0.339	0.348	0.360	0.326	0.313	0.123	0.136	0.236	0.243
NCI41	0.329	0.359	0.328	0.360	0.331	0.359	0.312	0.337	0.247	0.356	0.360	0.287	0.283	0.243	0.229	0.306	0.304
NCI47	0.282	0.302	0.279	0.302	0.285	0.304	0.269	0.287	0.255	0.301	0.310	0.242	0.260	0.202	0.210	0.248	0.237
NCI81	0.243	0.263	0.242	0.263	0.242	0.258	0.237	0.260	0.268	0.274	0.279	0.236	0.249	0.215	0.218	0.231	0.241
NCI83	0.279	0.293	0.279	0.293	0.280	0.291	0.269	0.289	0.260	0.288	0.306	0.257	0.257	0.235	0.245	0.256	0.253
H1	0.245	0.257	0.245	0.257	0.245	0.255	0.240	0.258	0.231	0.250	0.249	0.240	0.235	0.207	0.198	0.229	0.234
H2	0.591	0.614	0.591	0.614	0.582	0.612	0.590	0.598	0.601	0.626	0.611	0.557	0.551	0.548	0.556	0.574	0.579
A1	0.168	0.188	0.168	0.188	0.159	0.177	0.158	0.176	0.135	0.142	0.145	0.141	0.135	0.132	0.137	0.152	0.155
H3	0.654	0.668	0.654	0.668	0.654	0.661	0.663	0.667	0.651	0.670	0.654	0.643	0.614	0.584	0.591	0.590	0.604
D1	0.282	0.271	0.281	0.269	0.301	0.304	0.277	0.241	0.306	0.362	0.368	0.309	0.315	0.322	0.305	0.303	0.323
D2	0.638	0.595	0.639	0.596	0.603	0.580	0.598	0.585	0.571	0.582	0.591	0.498	0.548	0.587	0.588	0.504	0.499
D3	0.481	0.500	0.479	0.501	0.461	0.491	0.455	0.495	0.443	0.508	0.526	0.452	0.448	0.467	0.453	0.471	0.475
D4	0.480	0.489	0.479	0.489	0.490	0.492	0.457	0.461	0.457	0.497	0.509	0.458	0.455	0.465	0.477	0.408	0.417
P1	0.705	0.693	0.705	0.692	0.696	0.685	0.709	0.705	0.717	0.707	0.705	0.610	0.621	0.542	0.539	0.543	0.562
P2	0.682	0.671	0.679	0.670	0.653	0.667	0.674	0.671	0.647	0.661	0.664	0.634	0.623	0.535	0.525	0.465	0.475
P3	0.656	0.628	0.656	0.627	0.651	0.639	0.654	0.629	0.647	0.600	0.610	0.616	0.607	0.639	0.633	0.565	0.559
P4	0.655	0.634	0.657	0.636	0.639	0.629	0.645	0.657	0.597	0.599	0.576	0.651	0.640	0.537	0.548	0.614	0.617
C1	0.812	0.819	0.817	0.818	0.810	0.815	0.810	0.821	0.821	0.810	0.814	0.768	0.772	0.812	0.825	0.815	0.816
M1	0.480	0.481	0.468	0.474	0.454	0.469	0.450	0.465	0.458	0.466	0.492	0.365	0.371	0.430	0.441	0.420	0.428
M2	0.627	0.642	0.622	0.635	0.605	0.617	0.602	0.622	0.646	0.651	0.682	0.520	0.526	0.603	0.614	0.621	0.600
M3	0.796	0.803	0.793	0.804	0.812	0.817	0.758	0.783	0.773	0.820	0.824	0.725	0.711	0.753	0.767	0.800	0.803
ARQB	0.929	0.951	0.924	0.950	0.917	0.945	0.894	0.929	0.891	0.945	0.957	0.834	0.838	0.790	0.799	0.829	0.838

Table 11. Wilcoxon statistical test for the schemes in Table 10.

Min-Max

	GF ($\mathcal{K}_b$)	GF ($\mathcal{K}_f$)	AF ($\mathcal{K}_b$)	AF ($\mathcal{K}_f$)	TF ($\mathcal{K}_b$)	TF ($\mathcal{K}_f$)	PF ($\mathcal{K}_b$)	PF ($\mathcal{K}_f$)	fp-n ($\mathcal{K}_b$)	ECFP ($\mathcal{K}_b$)	ECFP ($\mathcal{K}_f$)	CT ($\mathcal{K}_b$)	CT ($\mathcal{K}_f$)	MK ($\mathcal{K}_b$)	MK ($\mathcal{K}_f$)	FS ($\mathcal{K}_b$)	FS ($\mathcal{K}_f$)	W/E/L
GF ($\mathcal{K}_b$)		=	=	=	=	=	>	=	=	=	=	>	>	>	>	>	>	7/9/0
GF ($\mathcal{K}_f$)	=		=	=	>	=	>	>	>	=	=	>	>	>	>	>	>	10/6/0
AF ($\mathcal{K}_b$)	=	=		=	=	=	>	=	=	=	=	>	>	>	>	>	>	7/9/0
AF ($\mathcal{K}_f$)	=	=	=		>	=	>	>	>	=	=	>	>	>	>	>	>	10/6/0
TF ($\mathcal{K}_b$)	=	<	=	<		<	=	=	=	=	<	>	>	>	>	>	>	6/6/4
TF ($\mathcal{K}_f$)	=	=	=	=	>		>	=	>	=	=	>	>	>	>	>	>	9/7/0
PF ($\mathcal{K}_b$)	<	<	<	<	=	<		<	=	<	<	>	>	>	>	>	>	6/2/8
PF ($\mathcal{K}_f$)	=	<	=	<	=	=	>		=	=	=	>	>	>	>	>	>	7/7/2
fp-n ($\mathcal{K}_b$)	=	<	=	<	=	<	=	=		<	<	>	>	>	>	>	=	5/6/5
ECFP ($\mathcal{K}_b$)	=	=	=	=	=	=	>	=	>		=	>	>	>	>	>	>	8/8/0
ECFP ($\mathcal{K}_f$)	=	=	=	=	>	=	>	=	>	=		>	>	>	>	>	>	9/7/0
CT ($\mathcal{K}_b$)	<	<	<	<	<	<	<	<	<	<	<		=	=	=	=	=	0/5/11
CT ($\mathcal{K}_f$)	<	<	<	<	<	<	<	<	<	<	<	=		=	=	=	=	0/5/11
MK ($\mathcal{K}_b$)	<	<	<	<	<	<	<	<	<	<	<	=	=		=	=	=	0/5/11
MK ($\mathcal{K}_f$)	<	<	<	<	<	<	<	<	<	<	<	=	=	=		=	=	0/5/11
FS ($\mathcal{K}_b$)	<	<	<	<	<	<	<	<	<	<	<	=	=	=	=		=	0/5/11
FS ($\mathcal{K}_f$)	<	<	<	<	<	<	<	<	=	<	<	=	=	=	=	=		0/6/10

The sign '>' denotes that row scheme outperforms column scheme, '<' denotes that column scheme outperforms row scheme and '=' denotes that row scheme and column scheme are statistically indistinguishable. W/E/L is Wins, Equal and Losses for each scheme.

Table 12. NHR ($k = 50$) for nine descriptors using $\mathcal{K}_b$ and $\mathcal{K}_f$ kernels.

Datasets	GF ($\mathcal{K}_b$)	GF ($\mathcal{K}_f$)	AF ($\mathcal{K}_b$)	AF ($\mathcal{K}_f$)	TF ($\mathcal{K}_b$)	TF ($\mathcal{K}_f$)	PF ($\mathcal{K}_b$)	PF ($\mathcal{K}_f$)	fp-n ($\mathcal{K}_b$)	ECFP ($\mathcal{K}_b$)	ECFP ($\mathcal{K}_f$)	CT ($\mathcal{K}_b$)	CT ($\mathcal{K}_f$)	MK ($\mathcal{K}_b$)	MK ($\mathcal{K}_f$)	FS ($\mathcal{K}_b$)	FS ($\mathcal{K}_f$)
NCI1	0.333	0.344	0.329	0.344	0.328	0.340	0.312	0.337	0.324	0.339	0.347	0.245	0.288	0.312	0.314	0.310	0.324
NCI109	0.298	0.311	0.298	0.311	0.298	0.308	0.280	0.305	0.297	0.301	0.319	0.213	0.241	0.276	0.279	0.285	0.300
NCI123	0.305	0.314	0.306	0.314	0.301	0.310	0.290	0.309	0.305	0.310	0.324	0.253	0.259	0.299	0.299	0.299	0.285
NCI145	0.323	0.335	0.323	0.335	0.320	0.332	0.301	0.327	0.312	0.320	0.339	0.322	0.316	0.297	0.299	0.306	0.308
NCI167	0.212	0.220	0.207	0.212	0.218	0.213	0.222	0.214	0.209	0.207	0.226	0.215	0.224	0.207	0.208	0.211	0.204
NCI220	0.317	0.318	0.315	0.318	0.322	0.312	0.316	0.310	0.315	0.323	0.306	0.322	0.329	0.306	0.310	0.312	0.308
NCI33	0.249	0.257	0.246	0.258	0.249	0.253	0.231	0.256	0.253	0.250	0.266	0.196	0.230	0.240	0.237	0.248	0.252
NCI330	0.354	0.369	0.351	0.368	0.346	0.362	0.331	0.357	0.351	0.377	0.399	0.303	0.301	0.336	0.339	0.342	0.352
NCI41	0.285	0.297	0.284	0.300	0.283	0.295	0.266	0.288	0.273	0.287	0.309	0.218	0.216	0.260	0.263	0.247	0.278
NCI47	0.304	0.318	0.302	0.318	0.310	0.317	0.285	0.309	0.292	0.308	0.327	0.225	0.241	0.284	0.286	0.268	0.299
NCI81	0.301	0.316	0.301	0.316	0.301	0.312	0.284	0.310	0.301	0.304	0.323	0.227	0.251	0.292	0.293	0.309	0.312
NCI83	0.307	0.316	0.308	0.316	0.301	0.313	0.285	0.308	0.300	0.303	0.322	0.238	0.248	0.293	0.292	0.297	0.289
H1	0.227	0.222	0.227	0.222	0.230	0.222	0.211	0.216	0.203	0.209	0.212	0.193	0.204	0.194	0.196	0.202	0.206
H2	0.308	0.291	0.308	0.291	0.309	0.295	0.305	0.287	0.301	0.314	0.311	0.253	0.241	0.274	0.279	0.249	0.254
A1	0.599	0.600	0.601	0.599	0.593	0.599	0.583	0.593	0.584	0.575	0.584	0.646	0.653	0.576	0.578	0.585	0.596
H3	0.540	0.540	0.540	0.540	0.546	0.545	0.537	0.549	0.517	0.500	0.496	0.525	0.509	0.489	0.484	0.486	0.495
D1	0.120	0.104	0.119	0.105	0.110	0.104	0.112	0.098	0.114	0.127	0.107	0.151	0.147	0.112	0.115	0.122	0.104
D2	0.236	0.238	0.236	0.238	0.243	0.238	0.245	0.237	0.236	0.226	0.226	0.288	0.245	0.246	0.252	0.268	0.239
D3	0.243	0.231	0.241	0.230	0.258	0.243	0.234	0.224	0.240	0.224	0.241	0.357	0.334	0.222	0.218	0.210	0.223
D4	0.361	0.358	0.361	0.358	0.366	0.361	0.367	0.353	0.364	0.356	0.353	0.435	0.417	0.367	0.366	0.348	0.370
P1	0.361	0.375	0.361	0.374	0.389	0.390	0.368	0.363	0.339	0.347	0.341	0.402	0.392	0.378	0.369	0.373	0.372
P2	0.289	0.287	0.288	0.287	0.310	0.296	0.289	0.280	0.269	0.293	0.278	0.311	0.297	0.306	0.300	0.285	0.291
P3	0.287	0.291	0.289	0.292	0.311	0.303	0.295	0.290	0.279	0.276	0.279	0.282	0.277	0.318	0.311	0.299	0.306
P4	0.305	0.304	0.305	0.304	0.333	0.322	0.316	0.306	0.301	0.285	0.288	0.351	0.322	0.344	0.357	0.346	0.371
C1	0.541	0.539	0.541	0.539	0.605	0.606	0.542	0.535	0.510	0.505	0.500	0.671	0.662	0.494	0.489	0.496	0.489
M1	0.320	0.332	0.319	0.332	0.314	0.326	0.325	0.329	0.319	0.341	0.356	0.261	0.249	0.297	0.301	0.274	0.300
M2	0.310	0.305	0.308	0.305	0.290	0.302	0.314	0.309	0.286	0.348	0.371	0.309	0.323	0.308	0.314	0.310	0.291
M3	0.330	0.323	0.328	0.322	0.300	0.314	0.349	0.331	0.341	0.401	0.419	0.277	0.268	0.314	0.324	0.325	0.320
ARQB	0.897	0.905	0.894	0.904	0.906	0.909	0.878	0.891	0.874	0.899	0.920	0.858	0.860	0.865	0.869	0.864	0.874

Table 13. Wilcoxon statistical test for the schemes in Table 12.

Min-Max

	GF ($\mathcal{K}_b$)	GF ($\mathcal{K}_f$)	AF ($\mathcal{K}_b$)	AF ($\mathcal{K}_f$)	TF ($\mathcal{K}_b$)	TF ($\mathcal{K}_f$)	PF ($\mathcal{K}_b$)	PF ($\mathcal{K}_f$)	fp-n ($\mathcal{K}_b$)	ECFP ($\mathcal{K}_b$)	ECFP ($\mathcal{K}_f$)	CT ($\mathcal{K}_b$)	CT ($\mathcal{K}_f$)	MK ($\mathcal{K}_b$)	MK ($\mathcal{K}_f$)	FS ($\mathcal{K}_b$)	FS ($\mathcal{K}_f$)	W/E/L
GF ($\mathcal{K}_b$)		=	=	=	=	=	=	=	>	=	=	=	=	>	>	>	=	4/12/0
GF ($\mathcal{K}_f$)	=		=	=	=	=	=	>	>	=	=	=	=	>	>	>	>	6/10/0
AF ($\mathcal{K}_b$)	=	=		=	=	=	=	=	>	=	=	=	=	=	=	=	=	1/15/0
AF ($\mathcal{K}_f$)	=	=	=		=	=	=	>	>	=	=	=	=	>	>	>	>	6/10/0
TF ($\mathcal{K}_b$)	=	=	=	=		=	>	=	>	=	=	=	=	>	>	>	>	6/10/0
TF ($\mathcal{K}_f$)	=	=	=	=	=		>	>	>	=	=	=	=	>	>	>	>	7/9/0
PF ($\mathcal{K}_b$)	=	=	=	=	<	<		=	=	=	=	=	=	=	=	=	=	0/14/2
PF ($\mathcal{K}_f$)	=	<	=	<	=	<	=		=	=	=	=	=	=	=	=	=	0/13/3
fp-n ($\mathcal{K}_b$)	<	<	<	<	<	<	=	=		=	<	=	=	=	=	=	=	0/9/7
ECFP ($\mathcal{K}_b$)	=	=	=	=	=	=	=	=	=		<	=	=	=	=	=	=	0/15/1
ECFP ($\mathcal{K}_f$)	=	=	=	=	=	=	=	=	>	>		=	=	=	>	>	=	4/12/0
CT ($\mathcal{K}_b$)	=	=	=	=	=	=	=	=	=	=	=		=	=	=	=	=	0/16/0
CT ($\mathcal{K}_f$)	=	=	=	=	=	=	=	=	=	=	=	=		=	=	=	=	0/16/0
MK ($\mathcal{K}_b$)	<	<	=	<	<	<	=	=	=	=	=	=	=		=	=	=	0/11/5
MK ($\mathcal{K}_f$)	<	<	=	<	<	<	=	=	=	=	<	=	=	=		=	=	0/10/6
FS ($\mathcal{K}_b$)	<	<	=	<	<	<	=	=	=	=	<	=	=	=	=		=	0/10/6
FS ($\mathcal{K}_f$)	=	<	=	<	<	<	=	=	=	=	=	=	=	=	=	=		0/12/4

The sign '>' denotes that row outperforms column descriptor, '<' denotes that column outperforms row descriptor and '=' denotes that row and column descriptors are statistically indistinguishable. W/E/L is Wins, Equal, and Losses for each scheme.

Table 14. ROC values for the nine methods for chemical compound classification.

Datasets	GF	AF	TF	PF	CT	RWK	WDK	FSG
CA+CM vs CI	**0.828**	0.822	0.815	0.803	0.809		0.817	0.765
CA vs CI	0.949	**0.950**	0.943	0.934	0.925		0.940	0.839
CA vs CM	0.834	0.833	0.830	0.822	0.826		**0.842**	0.810
MaleRats	**0.709**	0.708	0.705	0.708		0.632	0.697	0.626
FemaleRats	0.675	0.670	0.674	**0.688**		0.664	0.649	0.634
MaleMice	0.698	0.695	0.685	**0.716**		0.656	0.705	0.655
FemaleMice	**0.756**	0.753	0.736	0.733		0.645	0.691	0.673

Best performing scheme(s) for each classification problem is shown in bold.

perform other descriptors. Moreover, using frequency information in the kernel function ($\mathcal{K}_f$) leads to better results than just the binary presence/absence. An interesting observation is that although CT, FS and MK form the set of schemes that perform the worst among all the nine descriptors, fp-n is only slightly better than CT, MK or FS in terms of ARQB and statistically all the four are equivalent. This is not the case in classification where fp-n does significantly better than CT, MK and FS. Also the average performance of the AF, TF, and PF descriptors (as measured by AQRB) is higher than fp-n, CT, MK and FS as well.

7.6. Comparison with Published Results

In recent years many new descriptors and graph kernels have been introduced in the datamining literature and their classification performance has been successfully assessed. The performance assessment measure used in those studies is primarily area under the ROC curve. In Table 14 we compare the ROC results of GF, AF, TF, and PF with the results of recently introduced Cycles & Trees (CT) [25] , random-walk based graph kernels (RWK) [28], weighted decomposition kernels (WDK) [34] and Frequent subgraph based descriptors [18]. We use the length-differentiated Min-Max kernel ($\mathcal{K}_f^*$) for GF-based descriptors and its subsets. The results could only be compared for the common datasets with those used in these studies. We use the default misclassification cost factor (1.0) and do not optimize for regularization parameter in GF-based descriptors and its subsets. We compare our results with the best results (Gaussian version of intersection kernel described in [25]) of Cycles & Trees using misclassification cost of 1.0. In the case of WDK, the authors only report numbers with misclassification cost set to match the positive to negative compound ratio. They also optimize the regularization parameter. Hence we had to use best results from those numbers. For RWK, the code was provided to us by Mr. Kashima. We report RWK numbers for PTC dataset only as it was not possible to generate results for RWK for large aids dataset owing to the high computational complexity of the scheme. It can be observed from Table 14 that the GF descriptor outperforms CT, RWK, WDK and FSG for the majority of the datasets. Moreover, the best performing method consistently fall into one of the GF, AF, TF, or PF descriptors (except CA vs CM) despite the fact that no optimization performed on the SVM parameters. The average improvement of GF over the ROC values of WDK, CT, RWK, and FSG for the common datasets is 1.5%, 1.66%, 6% and 6.4% respectively.

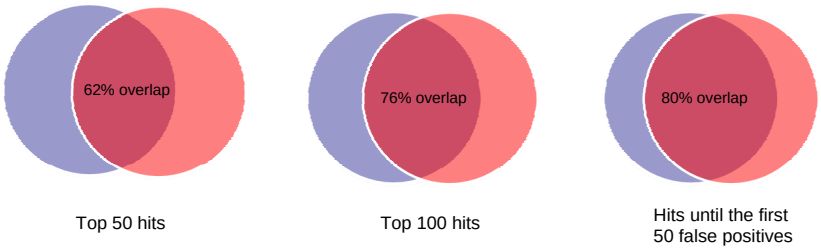

Fig. 1*a*. Overlap between sets of active compounds retrieved using GF descriptor ($\mathcal{K}_f$ kernel) and ECFP descriptor ($\mathcal{K}_f$ kernel) for classification.

8. Discussion and Conclusion

The work in this paper was primarily motivated by our desire to understand which aspects of the molecular graph are important in providing effective descriptor-based representations for the classification and retrieval tasks given the four design choices described in Section 4 (dataset specificity, fragment complexity, preciseness, and coverage) and the fact that no scheme leads to a descriptor space that is strictly superior (in terms of what it captures) to the rest of the schemes. Most of the descriptor spaces make some compromises along at least one of these dimensions. We believe that the experimental results presented in Section 7.5 provide some answers on the relative importance and impact of these design choices.

Specifically, the results comparing PF and fp-n, suggest that a precise representation is a key property and helps PF outperform fp-n even though the former utilizes only path-based fragments, whereas fp-n also uses fragments corresponding to cycles. Similarly, the results comparing GF against FS suggest that the 100% coverage of GF is a critical property as it helps outperform the FS approach. To ascertain this fact we decided to eliminate infrequent fragments of GF by applying support threshold similar to FS scheme on all datasets (data not shown). We found that the performance of the resulting classifier degrades as compared to GF. Thus, arbitrary support thresholds used to limit the number of fragments generated in graph mining deteriorates classification performance. Generating all fragments with support greater than or equal to one but having an upper limit on the size of fragments (GF) is a much better approach to classify chemical compounds. Also, the results comparing the schemes that utilize dataset specific fragment discovery approaches against the MK scheme show that relying on pre-identified fragments will lead to lower performance. Finally, the results comparing GF against AF, TF and PF show that everything else being the same, more complex fragments do lead to better results; however, these gains are not substantial.

Our results show that the GF and ECFP descriptor spaces that (nearly) satisfy all four of the desirable design choices, achieve the best results for both the classification and retrieval tasks. Moreover, these two descriptor-based representations are generally better than the state-of-the-art graph kernel approaches (RWK and WDK in Section 7.6) that operate directly on the compound's molecular graph. Furthermore, the advantage of the descriptor-based representation over the graph kernel approach is that the process of determining the simi-

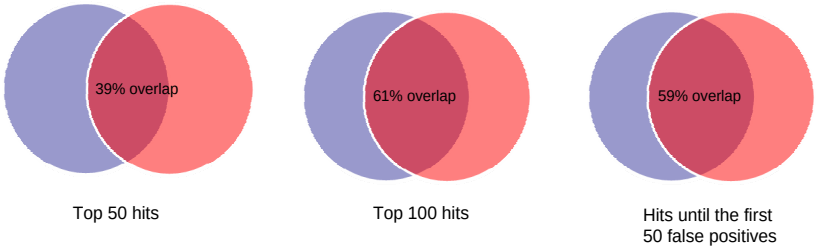

Fig. 1*b*. Overlap between sets of active compounds retrieved using GF descriptor ($\mathcal{K}_f$ kernel) and ECFP descriptor ($\mathcal{K}_f$ kernel) for ranked-retrieval.

larity between two compounds is decoupled from the actual descriptor space generation. To a large extent, this make the process of designing better descriptor spaces and similarity (kernel) functions independent of each other and thus much easier. However, a potential drawback is that it requires careful selection of the descriptor-based representation as well as the similarity function for the particular dataset and descriptor space, respectively.

In analyzing the compounds that were predicted to be positive or ranked higher in the classification or retrieval tasks, we found that there are considerable differences as to the true positives or hits that were identified by the GF and ECFP descriptor spaces. For example, Figure 1*a* shows how the true positives overlap[1] between the two schemes on three different subsets: (i) top 50 predictions, (ii) top 100 predictions, and (iii) all compounds until encountering the first 50 false positives. In the first subset, the overlap is only 62%, whereas for the third subset (which is usually contains more compounds than the other two), the overlap increases to 80%. A similar analysis is shown in Figure 1*b* for the retrieval task, and this time the overlap percentages are somewhat smaller, ranging from 39% to 61%. These overlap results show that there are considerable differences between the predictions and rankings produced by the two descriptor spaces. Thus, even though the overall performance of the GF and ECFP descriptors (as measured by ARQB) is quite similar, they tend to produce qualitative different results. This suggests that applying Fusion based techniques (that combine rankings obtained from different descriptor spaces) [48] to the rankings produced by GF and ECFP might lead to improvement in the performance over just the GF or ECFP ranked-retrieval results.

Acknowledgment

This work was supported by NSF EIA-9986042, IIS-0431135, NIH RLM008713A, ACI 0133464, the Army High Performance Computing Research Center contract number DAAD19-01-2-0014, and by the Digital Technology Center at the University of Minnesota.

[1] These overlap percentages correspond to the averages over the 28 problems.

References

[1] http://dtp.nci.nih.gov. *The Aids Antiviral Screen.*

[2] http://www.chemcomp.com. *Chemical Computing Group.*

[3] http://www.daylight.com. *Daylight Inc.*

[4] http://www.mdl.com. *MDL Information Systems Inc.*

[5] http://www.scitegic.com. *Scitegic Inc.*

[6] http://www.tripos.com. *Tripos Inc.*

[7] http://www.tripos.com. *Tripos Inc.*

[8] Mdl drug data report, version 2002.2. *MDL Information Systems Inc. San Leandro, CA.*

[9] pubchem.ncbi.nlm.nih.gov. *The PubChem Project.*

[10] www.chemaxon.com. *ChemAxon Inc.*

[11] www.predictive-toxicology.org.

[12] Food and drug administration orange book, 22nd edition. *U.S Food and Drug Administration, Washington D.C*, 2003.

[13] B. N. Ames, W. E. Durston, E. Yamasaki, and F. D. Lee. Carcinogens are mutagens: A simple test system combining liver homogenates for activation and bacteria for detection. *Proc. Natl. Acad. Sci.*, 70:2281–2285, 1973.

[14] J. M. Barnard and G. M. Downs. Chemical fragment generation and clustering software. *J. Chem. Inf. Comput. Sci.*, 37:141–142, 1997.

[15] J. M. Bland. An introduction to medical statistics. *(1995) 2nd edn. Oxford University Press.*

[16] H. Bohm and G. Schneider. Virtual screening for bioactive molecules. *Wiley-VCH, 2000.*

[17] R. Brown and Y. Martin. Use of structure-activity data to compare structure-based clustering methods and descriptors for use in compound selection. *J. Chem. Info. Model.*, 36(1):576–584, 1996.

[18] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis. Frequent substructure-based approaches for classifying chemical compounds. *IEEE TKDE.*, 17(8):1036–1050, 2005.

[19] J. L. Durant, B. A. Leland, D. R. Henry, and J. G. Nourse. Reoptimization of mdl keys for use in drug discovery. *J. Chem. Info. Model.*, 42(6):1273–1280, 2002.

[20] L. S. Gold and E. Zeiger. Handbook of carcinogenic potency and genotoxicity databases. *CRC Press*, 1997.

[21] M. Gribskov and N. Robinson. Use of receiver operating characteristic (roc) analysis to evaluate sequence matching. *Computational Chemistry*, 20:25–33, 1996.

[22] C. Helma, T. Cramer, S. Kramer, and L. D. Raedt. Data mining and machine learning techniques for the identification of mutagenicity inducing substructures and structure activity relationships of noncongeneric compounds. *J. Chem. Info. Comp. Sci.*, 44(4):1402–1411, 2004.

[23] J. Hert, P. Willet, and D. Wilton. New methods for ligand based virtual screening: Use of data fusion and machine learning to enchance the effectiveness of similarity searching. *J. Chem. Info. Model.*, (46):462–470, 2006.

[24] J. Hert, P. Willet, D. Wilton, P. Acklin, K. Azzaoui, E. Jacoby, and A. Schuffenhauer. Comparison of topological descriptors for similarity-based virtual screening using multiple bioactive reference structures. *Organic and Biomolecular Chemistry*, 2:3256–3266, 2004.

[25] T. Horvath, T. Grtner, and S. Wrobel. Cyclic pattern kernels for predictive graph mining. *SIGKDD*, pages 158–167, 2004.

[26] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. *PKDD*, pages 13–23, 2000.

[27] T. Joachims. Advances in kernel methods: Support vector learning, making large-scale svm learning practical. *MIT-Press, 1999.*

[28]H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs,. *ICML.*, 2003.

[29]L. Kier and L. Hall. Molecular structure description. *ic Press, 1999.*

[30]T. Kogej, O. Engkvist, N. Blomberg, and S. Moresan. Multifingerprint based similarity searches for targeted class compound selection. *J. Chem. Info. Model.*, 46(3):1201–1213, 2006.

[31]S. Kramer, L. D. Raedt, and C. Helma. Molecular feature mining in hiv data. *SIGKDD*, 2001.

[32]M. Kuramochi and G. Karypis. An efficient algorithm for discovering frequent subgraphs. *IEEE TKDE.*, 16(9):1038–1051, 2004.

[33]A. R. Leach. Molecular modeling: Principles and applications. *Prentice Hall, Englewood Cliffs, NJ, 2001.*

[34]S. Menchetti, F. Costa, and P. Frasconi. Weighted decomposition kernels. *ICML*, 2005.

[35]H. L. Morgan. The generation of unique machine description for chemical structures: a technique developed at chemical abstract services. *Journal of Chemical Documentation*, 5:107–113, 1965.

[36]S. Nijssen and J. N. Kok. A quickstart in frequent structure mining can make a difference. *SIGKDD*, 2004.

[37]G. W. Richards. Virtual screening using grid computing: the screensaver project. *Nature Reviews: Drug Discovery*, 1:551–554, July 2002.

[38]D. Rogers, R. Brown, and M. Hahn. Using extended-connectivity fingerprints with laplacian-modified bayesian analysis in high-throughput screening. *J. Biomolecular Screening*, 10(7):682–686, 2005.

[39]J. C. Saeh, P. D. Lyne, B. K. Takasaki, and D. A. Cosgrove. Lead hopping using svm and 3d pharmacophore fingerprints. *J. Chem. Info. Model.*, 45:1122–113, 2005.

[40]A. Srinivasan, R. D. King, S. H. Muggleton, and M. Sternberg. The predictive toxicology evaluation challenge. *IJCAI-97*, pages 1–6, 1997.

[41]N. Stiefl, I. A. Watson, K. Baumann, and A. Zaliani. Erg: 2d pharmacophore descriptor for scaffold hopping. *J. Chem. Info. Model.*, 46:208–220, 2006.

[42]S. J. Swamidass, J. Chen, J. Bruand, P. Phung, L. Ralaivola, and P. Baldi. Kernels for small molecules and the prediction of mutagenicity, toxicity and anti-cancer activity. *Bioinformatics*, 21(1):359–368, 2005.

[43]P. V.A., B. I.I., P. D.E., and Z. N.S. Novel descriptors of molecular structure in qsar and qspr studies. *QSAR and Molecular Modeling: Concepts, Computational tools and Biological Applications*, pages 51–52, Sanz F.

[44]V. Vapnik. Statistical learning theory. *John Wiley, 1998.*

[45]M. Vieth, M. G. Siegel, R. E. Higgs, I. A. Watson, D. H. Robertson, K. A. Savin, G. L. Durst, and P. A. Hipskind. Characteristic physical properties and structural fragments of marketed oral drug. *J. Med. Chem.*, 47(1):224–232, 2004.

[46]N. Wale and G. Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. *International Conference in Datamining. (ICDM)*, 2006.

[47]D. B. West. Introduction to graph theory. *Prentice Hall (2001).*

[48]M. Whittle, V. J. Gillet, and P. Willett. Enhancing the effectiveness of virtual screening by fusing nearest neighbor list: A comparison of similarity coefficients. *J. Chem. Info. Model.*, 44:1840–1848, 2004.

[49]P. Willett. Chemical similarity searching. *J. Chem. Info. Model.*, 38(6):983–996, 1998.

[50]M. Wrlein, T. Meinl, I. Fischer, and M. Philippsen. A quantitative comparison of the subgraph miners mofa, gspan, ffsm, and gaston. *PKDD*, 2005.

[51]X. Yan and J. Han. gspan: Graph-based substructure pattern mining. *ICDM*, 2002.

[52]N. S. Zefirov and V. A. Palyulin. Fragmental approach in qspr. *J. Chem. Info. Model.*, 42(5):1112–1122, 2002.

[53]Q. Zhang and I. Muegge. Scaffold hopping through virtual screening using 2d and 3d

similarity descriptors: Ranking, voting and consensus scoring. *J. Chem. Info. Model.*, 49:1536–1548, 2006.

Appendix A: Generation of GF Descriptors

Any frequent subgraph mining algorithm with a support threshold of one can be used to derive a set of bounded size GF descriptors. We tried to generate bounded size GF using existing frequent subgraph mining algorithms like FSG [32], Gaston [36] and gSpan [51]. But we faced considerable runtime and memory problems in the case of FSG and Gaston for large datasets with support threshold of one. Also, the current implementation of gSpan that we downloaded does not support an upper bound on the length of fragments. Moreover, frequent subgraph mining algorithms are not designed for this particular task at hand. Specifically, these algorithms spend a significant amount of time in support computation or calculation of embedding lists to speed up support computation [50]. But since all the possible bounded size subgraphs are generated in GF, every subgraph discovered is frequent and hence there is no need for support computations. For these reasons, in order to generate all connected graph fragments, we developed an algorithm that was inspired by the recursive technique for generating all the spanning trees of a graph G [47].

Consider an arbitrary edge e of G, and let $S_e(G)$ be the set of spanning trees of G that contain e and $S_{\neg e}(G)$ be the set of all spanning trees of G that do not contain e. It is easy to see that (i) $S_e(G) \cap S_{\neg e}(G) = \emptyset$ and (ii) $S_e(G) \cup S_{\neg e}(G)$ is equal to the set of all spanning trees of G, denoted by $S(G)$. Now, if $S(G/e)$ denotes an *edge contraction* operation (i.e., the vertices incident on e are collapsed together) then $S_e(G)$ can be obtained from $S(G/e)$ by adding e. If $G \backslash e$ denotes an *edge deletion* operation, then $S_{\neg e}(G)$ is nothing more than $S(G \backslash e)$. From the above observations we can come up with the following recurrence relation for generating $S(G)$

$$S(G) = \begin{cases} \emptyset, & \text{if } G \text{ does not have any edge} \\ eS(G/e) \cup S(G \backslash e), & \text{otherwise,} \end{cases} \tag{4}$$

where e is an arbitrary edge of G, and $eS(G/e)$ denotes the set of all spanning trees obtained by adding e to each spanning tree in $S(G/e)$.

The recurrence relation of Equation 4 can be used to generate all the connected graph fragments of a certain length l by modifying it in three different ways. These modifications are needed to ensure that (i) arbitrary graph fragments and cyclic fragments are generated (ii) the graph fragments that are returned are connected, and (iii) only all the fragments of length l are returned. The first objective can be achieved by simply changing the edge contraction operation to an edge deletion operation. The second can be achieved by imposing the constraint that the edge e must be incident on a vertex of G that was obtained via an edge deletion operation, if such a vertex exist. If G does not have any such vertex (i.e., it corresponds to the original graph), then e is selected in an arbitrary fashion. The length requirement can be ensured by terminating the recurrence relation when exactly l edges have been selected. In light of these modifications, the new recurrence relation that generates all the connected graph fragments of length l,

denoted by $F(G, l)$ is given by

$$F(G,l) = \begin{cases} \emptyset, & \text{if } G \text{ has fewer than } l \text{ edges or } l = 0 \\ eF(G\backslash e, l-1) \cup F(G\backslash e, l), & \text{otherwise,} \end{cases} \tag{5}$$

where e is satisfies the above constraints.

In order to identify isomorphic fragments in the same graph, we use canonical labelling [32] for every fragment generated from a molecular graph. The canonical labelling of every fragment can also be used to count the number of embeddings of a fragment in a molecular graph. Note that the recurrence relation above generates each fragment only once. Thus two isomorphic fragments in the same molecular graph differ by at least one edge. Also note that since the primary goal of this paper is compare different descriptor spaces, we did not compare the performance of our algorithm to the various frequent mining algorithms.