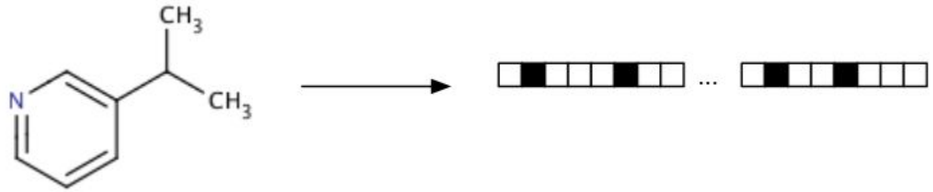
Molecular Fingerprinting

Drug Design Presentation

What are Fingerprints?

Idea : Apply a kernel to a molecule to generate a bit vector or count vector (less frequent)



Definition: Molecular descriptors that represent the structures of the molecules being compared.

- 1D descriptors include bulk properties and physicochemical parameters
  e.g., log P, molecular weight, polar surface area
- 2D descriptors include structural fragments or connectivity indices derived from the 2D representation of the molecule.
- 3D descriptors, such as molecular shape, are derived from 3D molecular structures
  i.e., 3D coordinates of the atoms in the molecule

Usages

- **Molecular Similarity**

- **Screening**

Substructure search is NP-Complete

but we can often detect the absence of a substructure much faster, often linear time.

screen: an "imperfect" algorithm

- can say pattern P «not in» M with 100% confidence but that can say P «in» M with lower confidence

- **other applications:**
  - property prediction [4]
  - synthesis design [5]
  - virtual screening [2,3,6]
  - cluster analysis [7,8]
  - molecular diversity analysis

Types of Fingerprints:

we focus on 2D molecular fingerprints

two types of molecular fingerprints are more widely used:

structural keys and hashed fingerprints

Structural keys

Structural keys

The structure of a molecule is encoded into a binary bit string

- Each bit corresponds to a pre-defined structural feature

  If the molecule has a feature, the bit position for it is 1. Otherwise 0.

Drawback: structural keys cannot encode features that are not pre-defined in the fragment library.

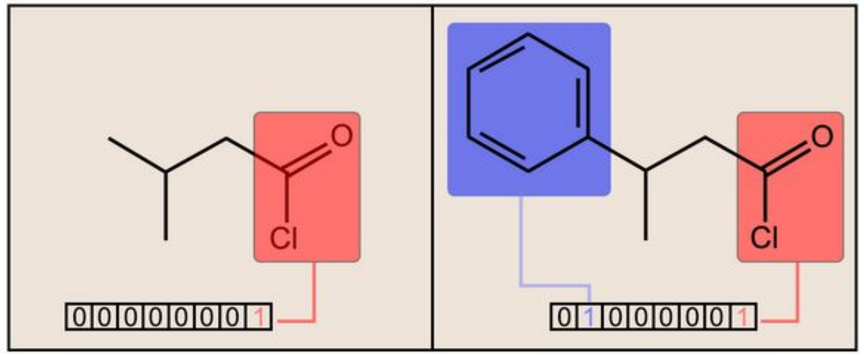


image source: https://chem.libretexts.org/Courses/Intercollegiate_Courses/Cheminformatics/06%3A_Molecular_Similarity/6.01%3A_Molecular_Descriptors

Similarity Measures

- Most common approach is Tanimoto similarity:

$$Tani(V_i, V_j) = \frac{V_i \bullet V_j}{\sum_b V_{ib} + \sum_b V_{jb} - V_i \bullet V_j}$$

- Dice similarity:

$$Sim(V_i, V_j) = \frac{2.0 * \sum_b \min(V_{ib}, V_{jb})}{\sum_b V_{ib} + \sum_b V_{jb}}$$

Structural keys – Examples

MACCS keys

aka the MDL keys, named after the company .

2 sets of keys (one with 960 keys and the other a subset of 166 keys),

public description available only for the shorter.(see here)

implemented in RDKit [20], OpenBabel [21,22], CDK [23,24], etc.

```
>>> from rdkit.Chem import MACCSkeys
>>> ms = [Chem.MolFromSmiles('CCOC'), Chem.MolFromSmiles('CCO'),
... Chem.MolFromSmiles('COC')]
>>> fps = [MACCSkeys.GenMACCSKeys(x) for x in ms]
>>> DataStructs.TanimotoSimilarity(fps[0],fps[1])
0.5
>>> DataStructs.TanimotoSimilarity(fps[0],fps[2])
0.538...
>>> DataStructs.TanimotoSimilarity(fps[1],fps[2])
0.214...
```

PubChem fingerprints

881-bit

used or similarity searching (interactively via the PubChem Homepage or by code via PUG-REST).

The fragment dictionary of the PubChem fingerprint is organized in seven sections:

ftp://ftp.ncbi.nlm.nih.gov/pubchem/specifications/pubchem_fingerprints.pdf

Structural keys – Daylight

- The presence/absence of elements

  if an element is common (e.g. N), several bits for "at least 1 N", "at least 2 N" etc.

- Electronic configurations, e.g. "sp3 carbon" or "triple-bonded nitrogen."

- Rings systems, e.g. cyclohexane, pyridine, or naphthalene.

- Common functional groups, e.g. alcohols, amines, hydrocarbons.

- Functional groups of special importance in a particular database. For example, a database of organo-metallic molecules might have bits assigned for metal-containing functional groups; in a drug database one might have bits for specific skeletal features such as steroids and barbiturates.

Structural keys – Drawbacks

Structural keys lack generality.

The choice of patterns included in the key has a critical effect on the search speed across the database:

a poor choice will cause many "false hits," which slows searching to a crawl.

The choice of patterns depends on the domain of the queries:

A structural key used by a group of pharmaceutical researchers might be nearly worthless to a group of petrochemical researchers.

Hashed Fingerprints

Hashed Fingerprints

Generated by enumerating all possible fragments that are not bigger than a certain size

- and then converting these fragments into numeric values using a “hash” function.
- These numeric values can be used to indicate bit positions in the hashed fingerprints.

all possible fragments in a molecule -> a very large number.

- Hashing them into values within a fixed range inevitably results in “bit collisions”

  no one-to-one correspondence between fragments and fingerprint bits (unlike structural keys).

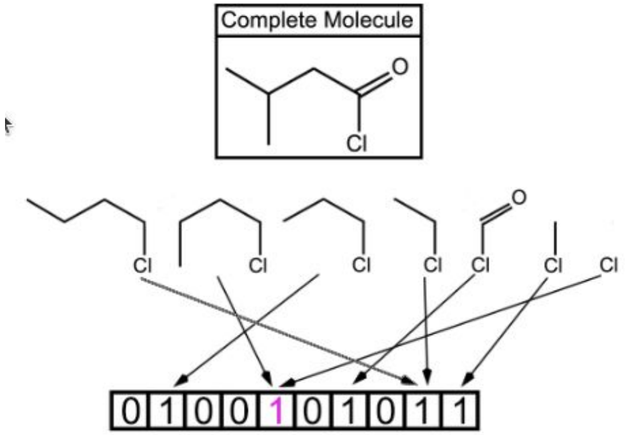


image source: https://chem.libretexts.org/Courses/Intercollegiate_Courses/Cheminformatics/06%3A_Molecular_Similarity/6.01%3A_Molecular_Descriptors

Hashed Fingerprints – Cont'd

A fingerprint is a boolean array (But in practice they're often used as counts)

- But there is no assigned meaning to each bit. (unlike structural keys)

Your own fingerprint is very characteristic of you, yet there is no meaning to any particular feature.

Similarly, a pattern's fingerprint characterizes the pattern, but the meaning of any particular bit is not well defined.

Hashed Fingerprints – Cont'd

Advantages over structural keys:

1. Does not require a pre-defined fragment library -> one fingerprint set for all databases and all types of queries.

2. More effective use is made of the bitmap.

Can be relatively "dense" (20-40% ones)

- Structural keys are usually very sparse

Types:

1. topological or path-based fingerprints
2. circular fingerprints

Path-based Fingerprints

Example - Daylight

The fingerprinting algorithm examines the molecule and generates the following:

- a pattern for each atom
- … for each atom and its nearest neighbors (plus the bonds that join them)
- … for each group of atoms and bonds connected by paths up to 2 bonds long
    - ... up to 3 bonds long
    - ... continuing, with paths up to 4, 5, 6, and 7 bonds long.

RDKit Fingerprinting is similar to Daylight.

Example: OC=CN

0-bond paths:	**C**	**O**	**N**
1-bond paths:	**OC**	**C=C**	**CN**
2-bond paths:	**OC=C**	**C=CN**	
3-bond paths:	**OC=CN**		

Path-based Fingerprints – Cont'd

Atom-pair descriptors [3] are available in several different forms. The standard form is as fingerprint including counts for each bit instead of just zeros and ones:

```
>>> ms = [Chem.MolFromSmiles('C1CCC1OCC'),Chem.MolFromSmiles('CC(C)OCC'),Chem.MolFromSmiles('CCOC
>>> fpgen = AllChem.GetAtomPairGenerator()
>>> pairFps = [fpgen.GetSparseCountFingerprint(x) for x in ms]
```

Because the space of bits that can be included in atom-pair fingerprints is huge, they are stored in a sparse manner. We can get the list of bits and their counts for each fingerprint as a dictionary:

```
>>> pairFps[-1].GetNonzeroElements()
{541732: 1, 558113: 2, 558115: 2, 558146: 1, 1606690: 2, 1606721: 2}
```

Unlike most other fingerprint types, descriptions of the bits are directly available:

```
>>> from rdkit.Chem.AtomPairs import Pairs
>>> Pairs.ExplainPairScore(558115)
(('C', 1, 0), 3, ('C', 2, 0))
```

The above means: C with 1 neighbor and 0 pi electrons which is 3 bonds from a C with 2 neighbors and 0 pi electrons

image source:
https://www.rdkit.org/docs/GettingStartedInPython.html

Circular Fingerprints

Considers the “circular” environment of each atom up to a given “radius” or “diameter”.

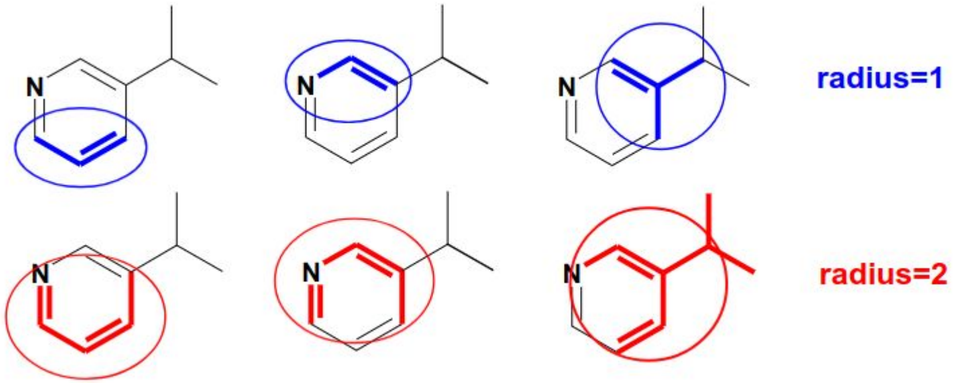


image source: https://www.rdkit.org/UGM/2012/Landrum_RDKit_UGM.Fingerprints.Final.pptx.pdf

- Typical radii: 0-3 bonds

Examples

- Morgan algorithm [28] - a method for solving the molecular isomorphism problem (i.e., how to identify identical molecules that have different atom numberings)

Morgan Fingerprints – Cont'd

Feature Definitions Used in the Morgan Fingerprints

Feature	SMARTS
Donor	`[$([N;!H0;v3,v4&+1]),$([O,S;H1;+0]),n&H1&+0]`
Acceptor	`[$([O,S;H1;v2;!$(*-*=[O,N,P,S])]),$([O,S;H0;v2]),$([O,S;-]),$([N;v3;!$(N-*=[O,N,P,S])]),n&H0&+0,$([o,s;+0;!$([o,s]:n);!$([o,s]:c:n))]`
Aromatic	`[a]`
Halogen	`[F,Cl,Br,I]`
Basic	`[#7;+,$([N;H2&+0][$([C,a]);!$([C,a](=O))]),$([N;H1&+0]([$([C,a]);!$([C,a](=O))])[$([C,a]);!$([C,a](=O))]),$([N;H0&+0]([C;!$(C(=O))])([C;!$(C(=O))])[C;!$(C(=O))])]`
Acidic	`[$([C,S](=[O,S,P])-[O;H1,-1])]`

ECFP Fingerprints

- extended-connectivity fingerprints (ECFPs) [27]

  - a variant of Morgan - computationally cheaper (source)

  ECFP2, ECFP4, ECFP6, etc. (the digit at the end = the maximum diameter)

  most commonly: ECFP4 and ECFP6

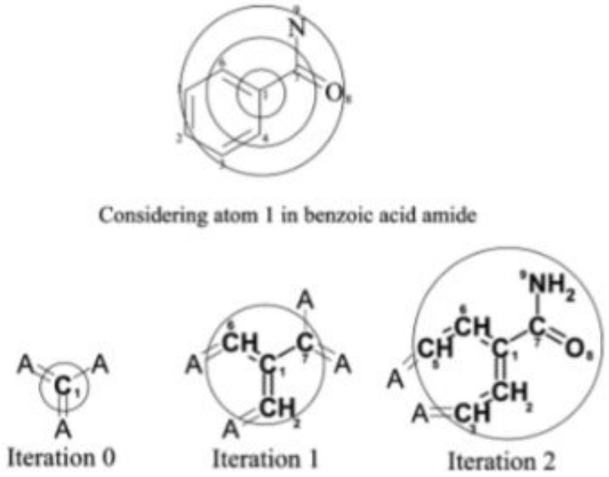


Figure 2. Illustration of the effect of iterative updating on the information represented by an atom identifier. Here, we consider atom 1 in benzoic acid amide. Each iteration has the effect of creating an identifier that represents larger and larger circular substructures around the central atom, as shown at the top of the figure. At iteration 0 (that is, the initial atom identifier), the atom only represents information about atom 1 and its attached bonds and can be represented by the substructure on the bottom left ("A" represents an atom of any type other than hydrogen). After one iteration, the identifier now contains information about atom 1's immediate neighbors, as shown in the bottom center substructure. After two iterations, the represented substructure has grown further, now fully incorporating the amide group as well as much of the aromatic ring, as shown in the bottom right.

image source: https://depth-first.com/articles/2019/01/11/extended-connectivity-fingerprints/

Circular Fingerprints – Cont'd

Other Examples:

circular fingerprints is functional-class fingerprints (FCFPs)

- a variation of ECFPs - encodes atom's roles (not atoms)

  First, each atom in the molecule is assigned a special code that represents one of the atom roles

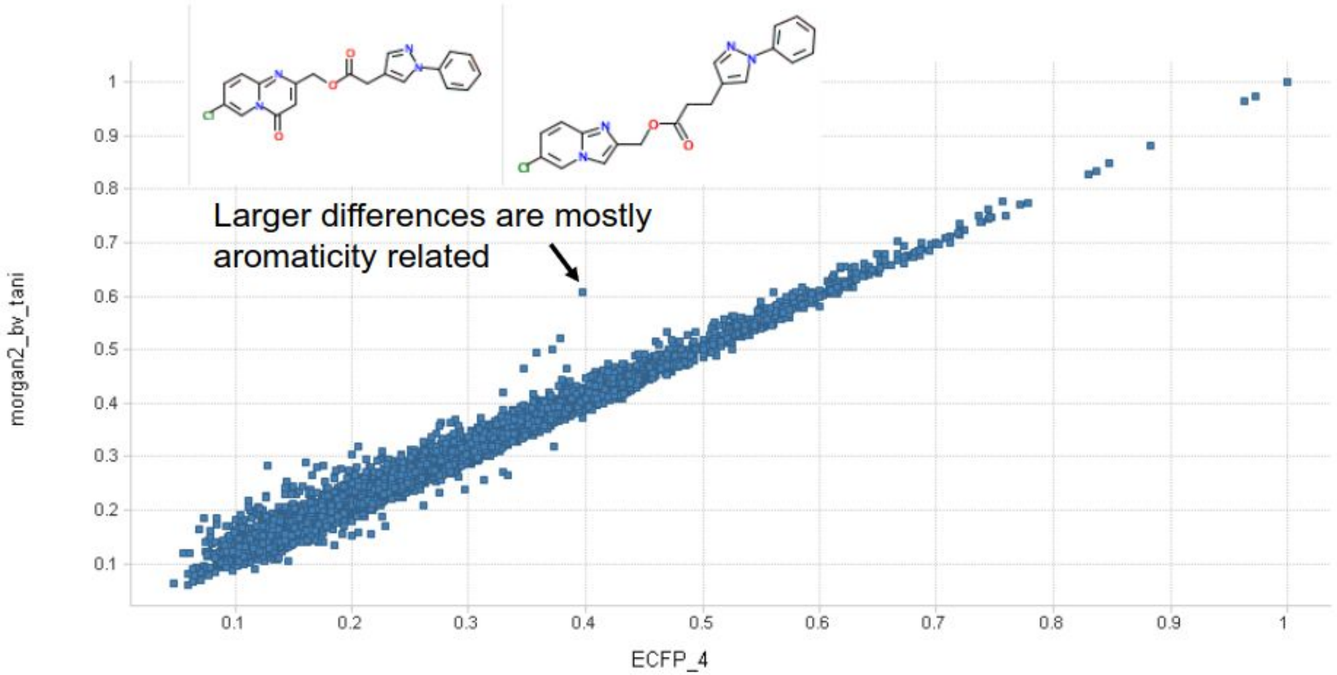
  (e.g., hydrogen-bond acceptor and donor, negatively or positively ionizable, aromatic, and halogen)

  Second, codes (not the atoms) are used to generate FCFPs, through the same process as ECFPs.

Circular Fingerprints – Cont'd

Morgan with radius r is roughly equal to ECFP[2 * r]

- RDKit Morgan2 vs PP ECFP4



- RDKit Morgan3 vs PP ECFP6 is similar

image source: https://depth-first.com/articles/2019/01/11/extended-connectivity-fingerprints/

Sources

https://chem.libretexts.org/Courses/Intercollegiate_Courses/Cheminformatics/06%3A_Molecular_Similarity/6.01%3A_Molecular_Descriptors

https://www.daylight.com/dayhtml/doc/theory/theory.finger.html

https://www.rdkit.org/UGM/2012/Landrum_RDKit_UGM.Fingerprints.Final.pptx.pdf

https://www.rdkit.org/docs/GettingStartedInPython.html#fingerprinting-and-molecular-similarity

https://depth-first.com/articles/2019/01/11/extended-connectivity-fingerprints/

Thank You