An Integrated Distance for Atoms

V. Estruch, C. Ferri, J. Hernández-Orallo and M.J. Ramírez-Quintana

DSIC, UPV, València, Spain.

{vestruch,cferri,jorallo,mramirez}@dsic.upv.es

Tenth International Symposium on Functional and Logic Programming.
FLOPS 2010

April 19-21, 2010
Sendai, Japan

Outline

1. Introduction
2. Related work
3. A new distance for atoms
4. Properties of distances
5. Discussion
6. Conclusions
7. Future work

Introduction

Distances over a set of objects:

- Set of tools and methods to work and analyse the objects therein.
- Potential applications in: debugging, termination, program analysis, and program transformation.

Functional and logic programming languages have many important applications as languages for object (knowledge) representation:

- LP: common formalism to represent (relational) knowledge
- FP: XML documents, related functional-alike structures....

Machine Learning + FLP: ILP (Progol) , IP, IFLP (FLIP) ..

Introduction

Distances over a set of objects:

- Set of tools and methods to work and analyse the objects therein.
- Potential applications in: debugging, termination, program analysis, and program transformation.

Functional and logic programming languages have many important applications as languages for object (knowledge) representation:

- LP: common formalism to represent (relational) knowledge
- FP: XML documents, related functional-alike structures....

Machine Learning + FLP: ILP (Progol) , IP, IFLP (FLIP) ..

Introduction

Distances over a set of objects:

- Set of tools and methods to work and analyse the objects therein.
- Potential applications in: debugging, termination, program analysis, and program transformation.

Functional and logic programming languages have many important applications as languages for object (knowledge) representation:

- LP: common formalism to represent (relational) knowledge
- FP: XML documents, related functional-alike structures....

Machine Learning + FLP: ILP (Progol) , IP, IFLP (FLIP) ..

Distance-Based Learning Methods

- The same technique can be applied to different sorts of data (with distance metric defined over them)
- Performance depends on the quality of the distance employed

One challenging case in machine learning is the distance between first-order atoms and terms.

- Can be used to represent different datatypes: lists,sets,...
- Especially suited for term-based or tree-based representations

Distance-Based Learning Methods

- The same technique can be applied to different sorts of data (with distance metric defined over them)
- Performance depends on the quality of the distance employed

One challenging case in machine learning is the distance between first-order atoms and terms.

- Can be used to represent different datatypes: lists,sets,...
- Especially suited for term-based or tree-based representations

Example (Motivation)

mol (H , s(s(Fe)) , [Au] , r(O,O))
mol (F , s(s(Fe)) , [Au] , r(O,O))
mol (H , s(s(Au)) , [Au] , r(O,O))
mol (H , s(Au) , [Ka,Nm,Fe] , r(O,O))
mol (H , s(Au) , [O] , r(O,O))
mol (H , s(Au) , [O] , r(Au,Fe))
mol (H , s(Au) , [O] , r(H,H))

Nienhuys-Cheng's distance

- Distance depends on their syntactic differences and on the positions where these differences take place.
  - Useful for ILP, XML documents, Ontologies
- A normalised function
  - Robust to noise, composability

J. Ramon et al. distance

- Considers repeated differences between atoms.
  - Common in terms
- Takes the syntactic complexity of differences into account
  - Refines the distances computed

Nienhuys-Cheng's distance

- Distance depends on their syntactic differences and on the positions where these differences take place.
  - Useful for ILP, XML documents, Ontologies
- A normalised function
  - Robust to noise, composability

J. Ramon et al. distance

- Considers repeated differences between atoms.
  - Common in terms
- Takes the syntactic complexity of differences into account
  - Refines the distances computed

This paper introduces a new distance for ground terms/atoms.

- Considers repetitions and syntactic complexity
- Preserves context-sensitivity, normalisation and composability.

Related Work

Niehuys-Cheng's distance

- Takes depth of symbols into account

- Given two ground terms/atoms $s = s_0(s_1, \ldots, s_n)$ and $t = t_0(t_1, \ldots, t_n)$, this distance is recursively defined as

$$d_N(s,t) = \begin{cases} 0, & \text{if } s = t \\ 1, & \text{if } \neg Compatible(s,t) \\ \frac{1}{2n}\sum_{i=1}^{n} d(s_i, t_i), & \text{otherwise} \end{cases}$$

Example (Nienhuys-Cheng's distance)

If $s = p(a,b)$ and $t = p(c,d)$ then
$d_N(s,t) = \frac{1}{4} \cdot (d(a,c) + d(b,d)) = \frac{1}{4}(1+1) = \frac{1}{2}$.

Related Work

Nienhuys-Cheng's distance

- Takes depth of symbols into account

- Given two ground terms/atoms $s = s_0(s_1, \ldots, s_n)$ and $t = t_0(t_1, \ldots, t_n)$, this distance is recursively defined as

$$d_N(s,t) = \begin{cases} 0, & \text{if } s = t \\ 1, & \text{if } \neg Compatible(s,t) \\ \frac{1}{2n}\sum_{i=1}^{n} d(s_i, t_i), & \text{otherwise} \end{cases}$$

Example (Nienhuys-Cheng's distance)

If $s = p(a,b)$ and $t = p(c,d)$ then
$d_N(s,t) = \frac{1}{4} \cdot (d(a,c) + d(b,d)) = \frac{1}{4}(1+1) = \frac{1}{2}$.

Related Work

Nienhuys-Cheng's distance

- Takes depth of symbols into account

- Given two ground terms/atoms $s = s_0(s_1, \ldots, s_n)$ and $t = t_0(t_1, \ldots, t_n)$, this distance is recursively defined as

$$d_N(s,t) = \begin{cases} 0, & \text{if } s = t \\ 1, & \text{if } \neg Compatible(s,t) \\ \frac{1}{2n} \sum_{i=1}^{n} d(s_i, t_i), & \text{otherwise} \end{cases}$$

Example (Nienhuys-Cheng's distance)

If $s = p(a,b)$ and $t = p(c,d)$ then
$d_N(s,t) = \frac{1}{4} \cdot (d(a,c) + d(b,d)) = \frac{1}{4}(1+1) = \frac{1}{2}$.

J. Ramon et al.'s distance

- Based on the syntactic differences wrt. their *lgg*

- An auxiliary function ($Size(t) = (F, V)$) is required to compute this distance
  - F counts the number of predicate and function symbols
  - V is the sum of the squared frequency of appearance of each variable in t
- Given two terms/atoms s and t this distance is

$$d_R(s,t) = [Size(s) - Size(lgg(s,t))] + [Size(t) - Size(lgg(s,t))]$$

J. Ramon et al.'s distance

- Based on the syntactic differences wrt. their *lgg*

- An auxiliary function ($Size(t) = (F, V)$) is required to compute this distance
  - F counts the number of predicate and function symbols
  - V is the sum of the squared frequency of appearance of each variable in t
- Given two terms/atoms s and t this distance is

$$d_R(s,t) = [Size(s) - Size(lgg(s,t))] + [Size(t) - Size(lgg(s,t))]$$

Example (J. Ramon et al.'s distance)

- If $s = p(a, b)$ and $t = p(c, d)$ and knowing that $lgg(s, t) = p(X, Y)$

$$
\begin{aligned}
&Size(s) = (3, 0)\\
&Size(t) = (3, 0)\\
&Size(lgg(s, t)) = (1, 2)\\
&d_R(s, t) = [(3, 0) - (1, 2)] + [(3, 0) - (1, 2)] =\\
&= (2, -2) + (2, -2) = (4, -4)
\end{aligned}
$$

A new distance for atoms

Definition of a new distance

- Complexity of the syntactic differences between the atoms
- Number of times each syntactic difference occurs
- Position (or context) where each difference takes place

Definition

(Syntactical differences between expressions) Let s and t be two expressions, the set of their syntactic differences, denoted by $O^\star(s,t)$, is defined as:

$$O^\star(s,t) \;=\; \{o \in O(s) \cap O(t) : \;\; \neg Compatible(s|_o, t|_o) \text{ and } Compatible(s|_{o'}, t|_{o'}), \forall o' \in Pre(o)\}$$

Example ($O^\star$)

$s = p(f(a), h(b), b)$, $t = p(g(c), h(d), d)$
$O^\star(s,t) = \{1, 2.1, 3\}$

Definition

(Size of an expression) Given an expression $t = t_0(t_1, \ldots, t_n)$, we define the function $Size'(t) = \frac{1}{4}Size(t)$ where

$$Size(t_0(t_1, \ldots, t_n)) = \begin{cases} 1, \; n = 0 \\ 1 + \frac{\sum_{i=1}^{n} Size(t_i)}{2(n+1)}, \; n > 0 \end{cases}$$

Example (Size)

$s = f(f(a), h(b), b)$
$Size(a) = Size(b) = 1,\; Size(f(a)) = Size(h(b)) = 1 + 1/4 = 5/4$
$Size(s) = 1 + (5/4 + 5/4 + 1)/8 = 23/16,\; Size'(s) = 23/64.$

Definition

(Context value of an occurrence) Let t be an expression. Given an occurrence $o \in O(t)$, the context value of o in t, denoted by $C(o;t)$, is defined as

$$C(o;t) = \begin{cases} 1, \; o = \lambda \\ 2^{Length(o)} \cdot \prod_{\forall o' \in Pre(o)} (Arity(t|_{o'}) + 1), \text{ otherwise} \end{cases}$$

Example (Context)

$t = p(g(c), h(d), d)$
$C(1;t) = 2 \cdot (3+1) = 8$
$C(2.1;t) = 2^2 \cdot (1+1) \cdot (3+1) = 32.$

Definition

(Function w) w simply associates weights to occurrences in such a way that the greater $C(o)$, the lower the weight o is assigned

$$\begin{array}{rcll} w: & O^{\star}(s,t) & \longrightarrow & \mathbb{R}^{+} \\ & o & \longmapsto & w(o) = \frac{3f_i(o)+1}{4f_i(o)}, \text{ where } i = \pi(o) \end{array}$$

Example (Function w)

$(O_2^{\star}(s,t), \leq) = \{3, 2.1\}$
$w(3) = 1$, $w(2.1) = 7/8$

Definition

(Distance between atoms) Let s and t be two expressions, the distance between s and t is,

$$d(s,t) = \sum_{o \in O^{\star}(s,t)} \frac{w(o)}{C(o)} \left(Size'(s|_o) + Size'(t|_o) \right)$$

Theorem

The ordered pair $(\mathcal{L},d)$ *is a bounded* $0 \leq d \leq 1$ *metric space .*

Proof.

For all expressions r, s and t in $\mathcal{L}$, the function d satisfies:

1. (Identity): $d(r,t) = 0 \Leftrightarrow r = t$.
2. (Symmetry): $d(r,t) = d(t,r)$.
3. (Triangular inequality): $d(r,t) \leq d(r,s) + d(s,t)$.
4. (Bounded distance): $0 \leq d(r,t) \leq 1$.

□

Example (1)

$s = f(a)$, $t = a$.

$$O^\star(s,t) = \{\lambda\}, C(\lambda) = 1$$

$$Size'(f(a)) = 5/16, Size'(a) = 1/4, w(\lambda) = 1$$

$$d(s,t) = \frac{1}{1}(Size'(s) + Size'(t)) = (\frac{5}{16} + \frac{1}{4})$$

Example (2)

$s = p(a, a),\ t = p(f(b), f(b)).$

$$O^\star(s,t) = \{1,2\}, C(1) = C(2) = 2 \cdot (2+1) = 6$$

$$Size'(a) = 1/4, Size'(f(b)) = 5/16$$

$$O^\star = O^\star_1(s,t), w(1) = 1 \text{ and } w(2) = 7/8$$

$$d(s,t) = \frac{1}{6}(\frac{1}{4} + \frac{5}{16}) + \frac{7}{48}(\frac{1}{4} + \frac{5}{16})$$

Properties of distances

Properties of distances

1. **Context Sensitivity**: it is the possibility of considering where the differences between two terms/atoms occur.
   - The distance between $p(a)$ and $p(b)$ should be greater than the distance between $p(f(a))$ and $p(f(b))$
2. **Normalisation**: a distance function d which returns (non-negative) real numbers can be easily normalised.
3. **Repeated differences**: this concerns the issue of handling repeated differences between terms/atoms properly.

   Consider $r = p(a, a)$, $s = p(b, b)$ and $t = p(c, d)$. Intuitively, r and s come nearer than r and t (or s and t), since r and s share that their (sub)terms (a and b, respectively) occur twice whereas no (sub)term is repeated in t.

Properties of distances between atoms

4. **Size of the differences**: is the complexity (the size) of the differences occurring when two terms/atoms are compared.

   Given the atoms $p(a)$, $p(b)$ and $p(f(c))$ then
   $d(p(a), p(b)) < d(p(a), p(f(c)))$,

5. **Handling variables**: Handling variables become a useful tool when part of the structure of an object is missing

6. **Composability**: The property of composability allows us to define distance functions for tuples by combining the distance functions defined over the basic types from which the tuple is constructed.

7. **Weights**: In some cases, it may be convenient to give higher or lower weights to some constants or function symbols,

   The distance between $f(a)$ and $f(b)$ could be greater than the distance between $f(c)$ and $f(d)$.

Advantages and drawbacks of several distances between terms/atoms

	Nienhuys-Cheng	J. Ramon et al.	Our distance
Context	Not always	Not always	Yes
Normalisation	Yes	Not easy	Yes
Repetitions	No	Yes	Yes
Size	No	Yes	Yes
Variables	Indirectly	Yes	Indirectly
Composability	Yes	Difficult	Yes
Weights	No	Yes	Indirectly

Discussion

A toy XML dataset with several car descriptions

- 8 Examples
- 12 Features, some of them not directly representable:
  - Photograph
  - Two numerical values

A representative extract from the XML dataset

```xml
<?xml version='1.0' ?>
<!DOCTYPE root SYSTEM "cars.dtd">
<root>
  <car>
    <company> Chevrolet </company>
    <model> Corvette </model>
    <certifications> E3 </certifications>
    <certifications> D52 </certifications>
    <certifications> RAC </certifications>
    <features>
      <color> red </color>
      <brake> abs </brake>
      <power> 250 </power>
      <airbag>
        <front> full </front>
        <rear> mid </rear>
      </airbag>
      <engine>
        <type> diesel </type>
        <turbo> yes </turbo>
      </engine>
    </features>
    <baseprice> 60,000 </baseprice>
    <photo> ChevCorv.jpg </photo>
  </car>
  . . .
</root>
```

An equivalent term-based representation of the XML dataset

1	car(Ford,Ka,cert([E3]),feats(75, red,abs,airbag(full,mid),motor(gas,no)), 9000, ChevKaG.jpg)
2	car(Ford,Ka,cert([E3]),feats(80, red,abs,airbag(full,mid),motor(diesel,yes)), 10000, ChevKaD.jpg)
3	car(Chev,Corv,cert([E3]),feats(250,red,abs,airbag(full,mid),motor(gas,no)), 60000, ChevCorv.jpg)
4	car(Ford,Ka,cert([E3]),feats(125, blue,abs,airbag(mid,mid),motor(diesel,yes)), 10000, ChevKaD2.jpg)
5	car(Ford,Ka,cert([E3]),feats(125, blue,abs,airbag(full,full),motor(diesel,yes)), 10500, ChevKa3.jpg)
6	car(Ford,Ka,cert([E3]),feats(125, blue,abs,airbag(extra,no),motor(diesel,yes)), 11000, ChevKaD4.jpg)
7	car(Chev,Xen,cert([D52, RAC, H5]),feats(300, red,abs,airbag(full,mid),motor(gas,no)), 70000, ChevXen.jpg)
8	car(Chev,Prot,cert([RAC]),feats(300, red,abs,airbag(full,mid),motor(gas,no)), 60000, ChevProt.jpg)

Example (Position of Differences)

- Cars 1, 2, and 3
  - `car(Ford,Ka,cert([E3]),feats(...,motor(gas,no)), 9000, ...`
  - `car(Ford,Ka,cert([E3]),feats(...,motor(diesel,yes)), 10000, ...)`
  - `car(Chev,Corv,cert([E3]),feats(...,motor(gas,no)), 60000, ...)`
- Car 1 looks more similar to car 2 than 1 to 3
  - Both pairs of cars $(1, 2)$ and $(1, 3)$ have an identical number of differences
- Differences at top positions in the atoms must be more important than differences at inner positions

Comparison

Distance	
Nienhuys-Cheng	√
J. Ramon et al.	×
Our distance	√

Example (Position of Differences)

- Cars 1, 2, and 3
  - `car(Ford,Ka,cert([E3]),feats(...,motor(gas,no)), 9000, ...`
  - `car(Ford,Ka,cert([E3]),feats(...,motor(diesel,yes)), 10000, ...)`
  - `car(Chev,Corv,cert([E3]),feats(...,motor(gas,no)), 60000, ...)`
- Car 1 looks more similar to car 2 than 1 to 3
  - Both pairs of cars $(1, 2)$ and $(1, 3)$ have an identical number of differences
- Differences at top positions in the atoms must be more important than differences at inner positions

Comparison

Distance	
Nienhuys-Cheng	√
J. Ramon et al.	×
Our distance	√

Flexible weights

- A context-sensitive distance allows us to indirectly use the position in the atom/term in order to set different levels of importance for every trait of the car
  - Moving the trait *colour* to a higher position in the atom implies that differences involving this attribute become more meaningful
- Artificial constructors allow us to reduce the importance of a trait
  - A nested expression $(art(art(art(Ford))))$ would decrease the importance of the trait *company*

Example (Size of differences)

- Cars 3, 7, and 8
  - `car(Chev,Corv,cert([E3]),...)`
  - `car(Chev,Xen,cert([D52, RAC, H5]),...)`
  - `car(Chev,Prot,cert([RAC]),...)`
- Cars 3 and 8 seem to be the most similar
  - They have only one certification while 7 has three
- The size of the differences must be taken into account

Comparison

Distance	
Nienhuys-Cheng	×
J. Ramon et al.	√
Our distance	√

Example (Size of differences)

- Cars 3, 7, and 8
  - `car(Chev,Corv,cert([E3]),...)`
  - `car(Chev,Xen,cert([D52, RAC, H5]),...)`
  - `car(Chev,Prot,cert([RAC]),...)`
- Cars 3 and 8 seem to be the most similar
  - They have only one certification while 7 has three
- The size of the differences must be taken into account

Comparison

Distance	
Nienhuys-Cheng	×
J. Ramon et al.	√
Our distance	√

Example (Repeated Differences)

- Cars 4, 5, and 6
  - `car(Ford,Ka,cert([E3]),feats(125, blue,abs,airbag(mid,mid),...)`
  - `car(Ford,Ka,cert([E3]),feats(125, blue,abs,airbag(full,full),...)`
  - `car(Ford,Ka,cert([E3]),feats(125, blue,abs,airbag(extra,no),...)`
- Cars 4 and 5 seem to be the most similar
  - 4 and 5 have a homogeneous airbag equipment
- Repeated differences must be considered

Comparison

Distance	
Nienhuys-Cheng	×
J. Ramon et al.	√
Our distance	√

Example (Repeated Differences)

- Cars 4, 5, and 6
  - `car(Ford,Ka,cert([E3]),feats(125, blue,abs,airbag(mid,mid),...)`
  - `car(Ford,Ka,cert([E3]),feats(125, blue,abs,airbag(full,full),...)`
  - `car(Ford,Ka,cert([E3]),feats(125, blue,abs,airbag(extra,no),...)`
- Cars 4 and 5 seem to be the most similar
  - 4 and 5 have a homogeneous airbag equipment
- Repeated differences must be considered

Comparison

Distance	
Nienhuys-Cheng	×
J. Ramon et al.	√
Our distance	√

Composability

- We have 3 special features: 2 numerical, 1 photograph
  - we can compute the distances for these features, getting three scalar values
- We can compose atom with non-atom representations (such as the picture) constructing a tuple
  - J. Ramon et al.'s distance with the rest, we have as a result a pair such as (n, m). Difficult to combine with the other distances
  - Nienhuys-Cheng's distance and ours can handle the whole XML description

An equivalent tuple-based representation of the atom representation

1	⟨ 75, 9000, ChevKaG.jpg, car(Ford,Ka,cert([E3]),...) ⟩
2	⟨ 80, 10000, ChevKaD.jpg, car(Ford,Ka,cert([E3]),...)⟩
3	⟨ 250, 60000, ChevCorv.jpg, car(Chev,Corv,cert([E3]),...)⟩
4	⟨ 125, 10000, ChevKaD2.jpg, car(Ford,Ka,cert([E3]),...)⟩
5	⟨ 125, 10500, ChevKa3.jpg, car(Ford,Ka,cert([E3]),...)⟩
6	⟨ 125, 11000, ChevKaD4.jpg, car(Ford,Ka,cert([E3]),...)⟩
7	⟨ 300, 70000, ChevXen.jpg, car(Chev,Xen,cert([D52, RAC, H5]),...)⟩
8	⟨ 300, 60000, ChevProt.jpg, car(Chev,Prot,cert([RAC]),...)⟩

Conclusions

- We have presented a new distance for ground terms/atoms which integrates the most remarkable traits in Nienhuys-Cheng's and J. Ramon et al.'s proposals
  - Context-sensitivity
  - Complexity
  - Repeated differences
  - Composability
- Direct applications in machine learning and inductive programming (ILP)
- Indirect applications in other areas of logic and functional programming : Debugging, termination, program analysis, and program transformation.

Future Work

- Considering weights directly (now by using dummy function symbols)
- Handling variables directly (as J. Ramon et al.'s distance does)
- Improving distances for nested data types (e.g. sequences of sets, or lists of lists, etc.).
- Implementing the distance to conduct experiments in ML or other areas