N-gram Graphs: A generic machine learning tool in the arsenal of NLP, Video Analysis and Adaptive Systems. (Part II)

George Giannakopoulos[1]

[1]University of Trento, Italy
ggianna@disi.unitn.it

April 26, 2010

In the previous episode...

- N-gram Graphs and Operators
- Richer information
- Domain agnostic
- Generic applicability
- State-of-the-art performance in summary evaluation
- Promising for language-independent summarization
- Usable in classification, clustering, record linkage

This episode

- Representing behavior (using Optical Flow Proximity Graphs)

This episode

- Representing behavior (using Optical Flow Proximity Graphs)
- Combining N-gram Graphs with the Vector Space

This episode

- Representing behavior (using Optical Flow Proximity Graphs)
- Combining N-gram Graphs with the Vector Space
- User Modeling with N-gram Graphs

This episode

- Representing behavior (using Optical Flow Proximity Graphs)
- Combining N-gram Graphs with the Vector Space
- User Modeling with N-gram Graphs
- The JINSECT toolkit: An open source LGPL toolkit for N-gram Graphs

Outline

Behavior Recognition

Examples

- Assistive Environment

Behavior Recognition

Examples

- Assistive Environment
- Super Market – Mall

Behavior Recognition

Examples

- Assistive Environment
- Super Market – Mall
- Parking Lot

Behavior Recognition

Examples

- Assistive Environment
- Super Market – Mall
- Parking Lot
- Vending Machines – ATMs

Behavior Recognition

Examples

- Assistive Environment
- Super Market – Mall
- Parking Lot
- Vending Machines – ATMs
- Traffic

Behavior Recognition

Examples

- Assistive Environment
- Super Market – Mall
- Parking Lot
- Vending Machines – ATMs
- Traffic
- Sports

Optical Flow

Image from http://api.ning.com/files/DPSX6QXHN*m77We5ozsvlC1V7uw5qyicb90jUaDEda2vMbj*cnWXOm9T8YtCG61DUl2ijCFR1n8OfnvFHaOjWokU5EXwtKxE/sam200.jpg

Behavior Recognition and Video Indexing

The Method[a]

[a] In collaboration with **Panagiota Antonakaki, NCSR Demokritos**.

Behavior Recognition and Video Indexing

The Method[a]

[a] In collaboration with **Panagiota Antonakaki, NCSR Demokritos**.

- No a priori information required

Behavior Recognition and Video Indexing

The Method[a]

[a] In collaboration with **Panagiota Antonakaki, NCSR Demokritos**.

- No a priori information required
- No preprocessing steps required

Behavior Recognition and Video Indexing

The Method[a]

[a] In collaboration with **Panagiota Antonakaki, NCSR Demokritos**.

- No a priori information required
- No preprocessing steps required
- Only optical flow for feature vector calculation

Representing Behavior — Variations

Proposed Methods

1. Whole frame representation using graphs (Optical Flow Proximity Graphs - OFPGs)

Representing Behavior — Variations

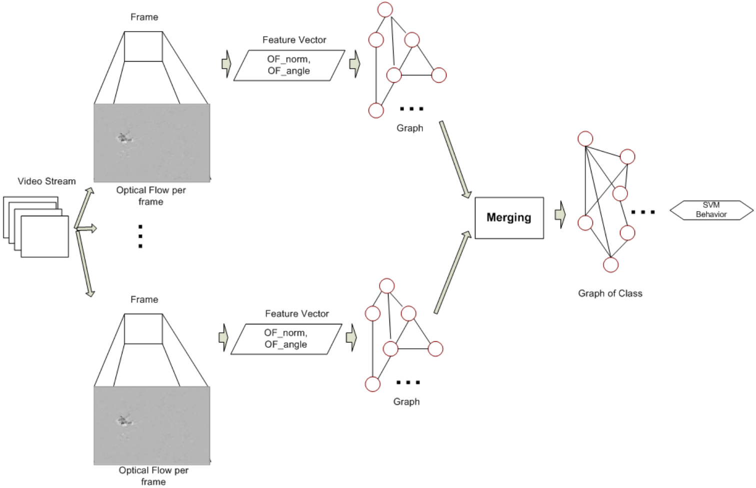
Proposed Methods

1. Whole frame representation using graphs (Optical Flow Proximity Graphs - OFPGs)
2. Segmentation and representation using hierarchy of graphs (Symbolic)

Outline

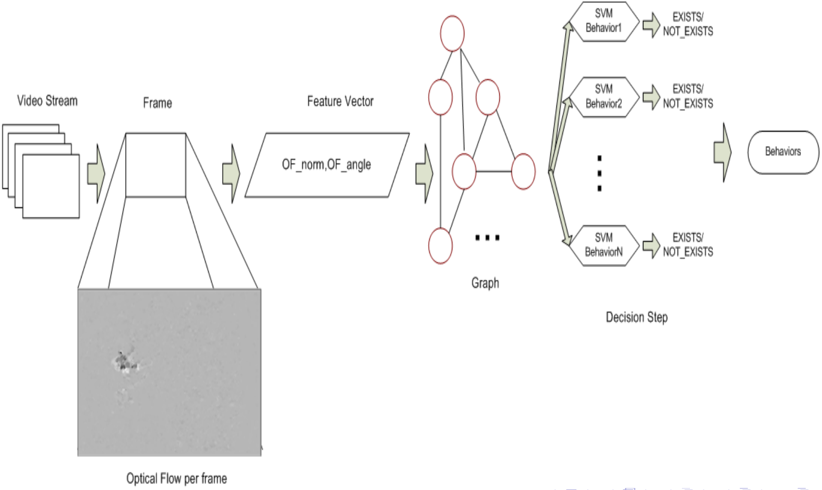
Whole Frame Representation (1)

Training Step



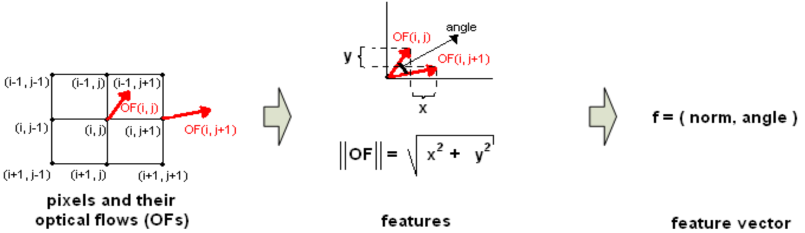
Whole Frame Representation (2)

Testing Step



Graph Representation from Vectors

Extraction of feature vector.



$$f = (OF_{norm}, OF_{angle}) \tag{1}$$

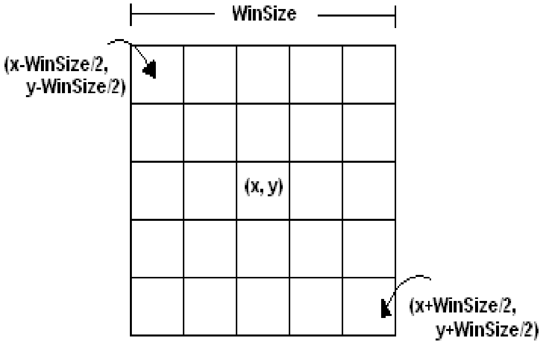
Features

$$abs = getBinForValue("abs", \sqrt{xVector^2 + yVector^2})$$

$$angle = getBinForValue("angle", (tan(xVector, yVector) + \pi) * 180/\pi)$$

where *getBinForValue* is a function that returns the name of the bin (quantization)

Parametrically Determined Window



What is Used?

Using N-gram Graph Operators

- Update operator

What is Used?

Using N-gram Graph Operators

- Update operator
- Comparison operator

What is Used?

Using N-gram Graph Operators

- Update operator
- Comparison operator
- Intersection operator

What is Used?

Using N-gram Graph Operators

- Update operator
- Comparison operator
- Intersection operator
- All-not-in or delta operator

Noise in Data

Reasons for removal

- Background noise due to camera

Noise in Data

Reasons for removal

- Background noise due to camera
- Classification lies in the differences between classes

Noise in Data

Reasons for removal

- Background noise due to camera
- Classification lies in the differences between classes

$$gNoise = intersection_j\{G_{c_j}\}, 1 \leq j \leq N$$

for *each* G_{c_j} **do**
  NoiselessClassGraph = NoiselessClassGraph.allNotIn(gNoise);
end

Outline

1. Evolution of N-gram Graphs to Video Analysis
   - The Optical Flow Proximity Graph
   - Whole Frame
   - Operators Revisited: Complexity
   - Hierarchy in Graphs
2. Modeling User Preferences
   - Overview of Senses
   - Representation
   - Overview of Solution
   - Data and Experiments
3. JINSECT: A Toolkit for All
   - Overview
   - Why use it?
4. Closing
   - Summary and the Future
   - Appendix

What is the Complexity of Trivially Implemented Graph Operators?

- Extraction from Source (of size N, m dimensions):

$$O({D_{\text{win}}}^{m} \times N)$$

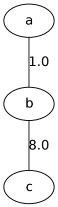
- Similarities $|G_m| = min(|G_1|, |G_2|), |G^M| = max(|G_1|, |G_2|)$
  - Size Similarity: $O(1)$
  - Containment and Value Similarity: $O(|G_m||G^M|)$
- Update, Merge: $O(|G_m||G^M| + |G_m|c)$

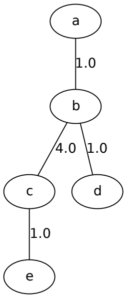
Improved Complexity of Graph Operators?

- Using hash for nodes (or edges)
- Quick search - slower insert
- Extraction from Text (of length N): $D_{\text{win}} \times N$
- Similarities $|G_m| = min(|G_1|, |G_2|), |G^M| = max(|G_1|, |G_2|)$
  - Size Similarity: $O(1)$
  - Containment and Value Similarity: $O(|G_m| \log |G^M|)$
  - Update, Merge: $O(|G_m| \log |G^M| + |G_m| c)$

N-gram Graph – Value Similarity

Example





Result: $\frac{\frac{1.0}{1.0}}{4} + \frac{\frac{4.0}{8.0}}{4} = \frac{1}{4} + \frac{1}{8} = 0.375$

Space Complexity and Related Considerations

- Vertices can be burdensome
- Edges can be burdensome
- Operators copying graphs
- Indexing increases memory requirement

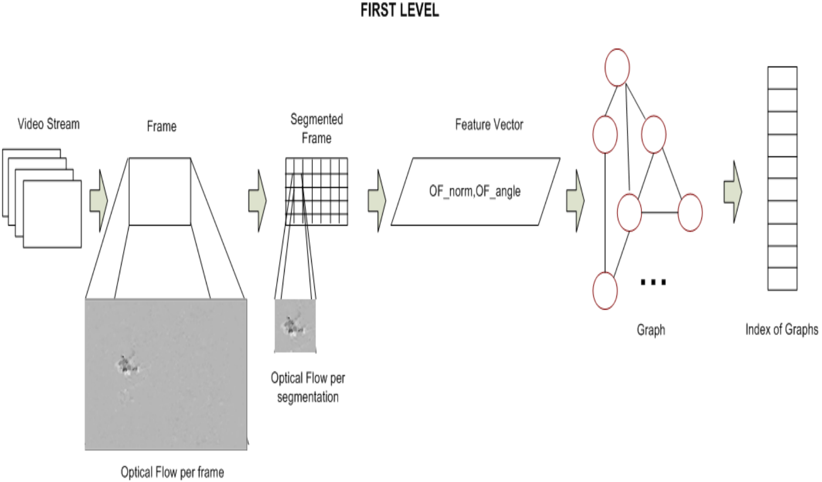
Space Complexity and Related Considerations

- Vertices can be burdensome
- Edges can be burdensome
- Operators copying graphs
- Indexing increases memory requirement
- Serialization

Outline

Segmented Frame Representation — Hierarchy (1)

Training Step

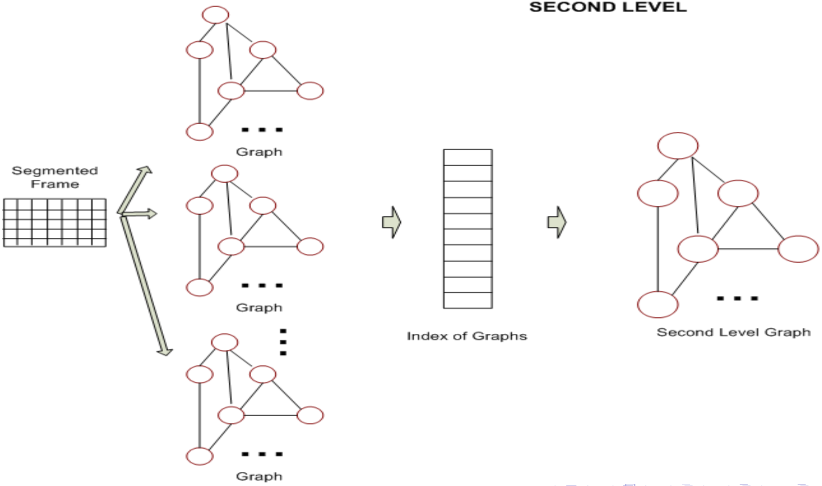


Searching and Updating the Index

Data: CurrentGraph, IndexOfGraphs
Result: informed IndexOfGraphs
dgName = null;
for *each Graph in set of graphs of IndexOfGraphs* **do**
  compute similarity between CurrentGraph and Graph;
  if $similarity \geq maxForMerging$ **then**
    dgName = name of the Graph;
  else if $similarity \geq minForMerging$ **then**
    dgName = name of the Graph;
    Graph = result of merging CurrentGraph and Graph;
  else
    CurrentGraph = result of removal of Graph from CurrentGraph
  end
end
if $dgName = null$ **then**
  assign a new name to the CurrentGraph and add CurrentGraph and name in the index;
end

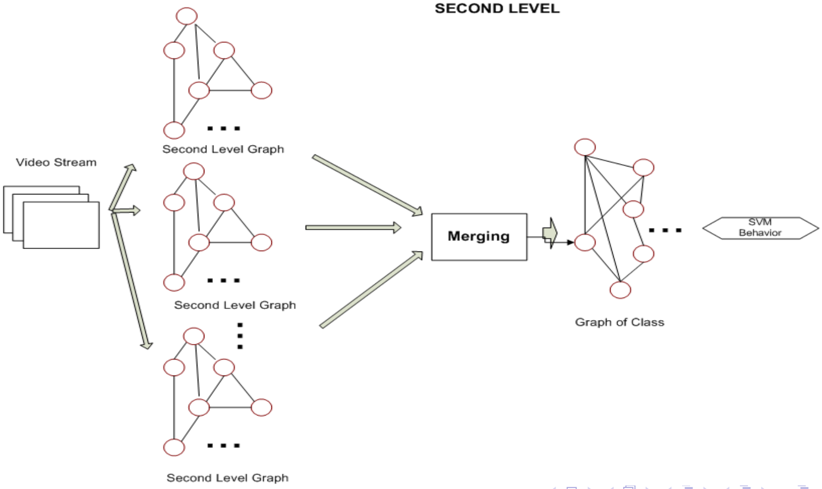
Segmented Frame Representation — Hierarchy (2)

Training Step



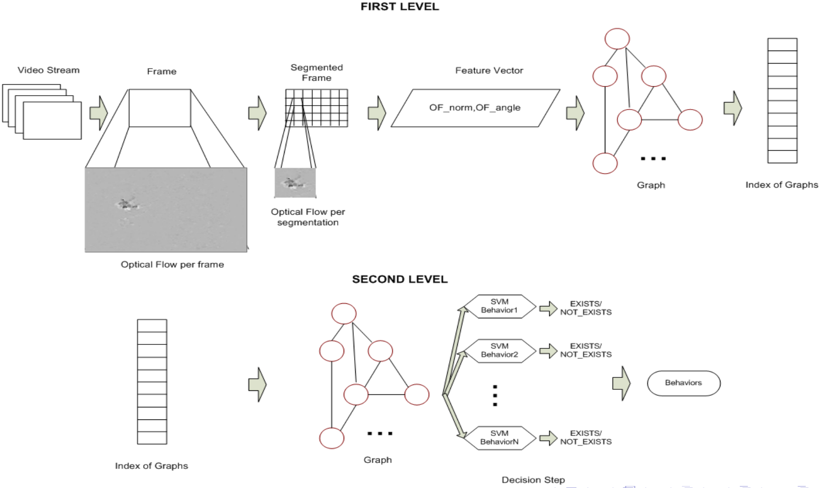
Segmented Frame Representation — Representing a Class

Training Step



Segmented Frame Representation — Testing

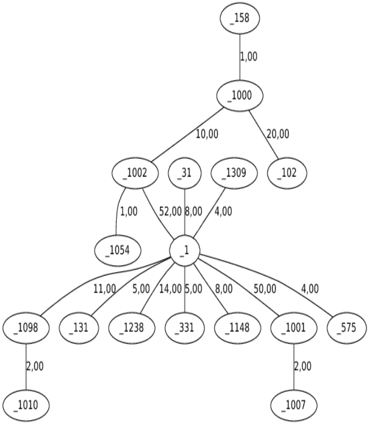
Testing Step



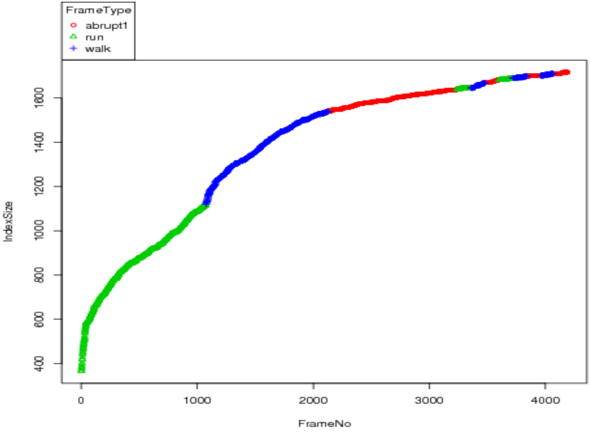
Frame as Symbols Example

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	2	1	1	1	1	1	2	1	1	1	1
1	1	3	3	1	1	1	1	1	1	1	1	3	3	3	1	4	2	1	1	1	1	1	1	2	1	3	1
1	1	3	1	2	1	1	1	1	1	1	1	3	3	1	2	3	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1
1	1	1	1	1	1	2	1	1	2	5	1	1	1	1	6	1	2	1	1	1	1	1	1	6	1	2	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	3	1	1	1	4	1	1
1	1	2	1	1	1	1	6	1	1	1	3	1	1	1	1	1	1	1	1	1	6	1	1	2	1	1	1
1	1	1	1	1	1	1	1	1	1	1	2	2	5	1	1	2	1	1	1	1	1	1	1	1	2	1	1
1	1	1	1	1	1	1	2	6	1	1	1	1	1	1	1	1	1	1	2	1	3	4	1	7	1	1	1
1	1	1	1	1	1	1	1	1	1	1	7	1	5	1	1	1	4	1	1	1	1	7	1	8	1	1	1
1	1	1	1	2	1	1	2	1	1	1	7	1	6	1	1	9	1	1	1	1	1	4	8	4	1	1	2
1	1	1	1	1	1	1	1	1	1	1	1	10	2	10	1	2	11	1	1	1	12	13	14	1	10	1	1
1	1	14	14	1	1	14	2	1	1	1	1	1	1	1	14	2	1	1	1	1	3	14	12	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	12	15	6	1	16	1	1	1	1	1	1	12	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	10	11	10	10	1	16	10	11	10	1	1	1	1
1	10	1	12	1	1	15	1	10	10	1	1	1	10	10	1	6	17	17	14	14	1	11	11	1	1	1	18
1	1	1	1	10	10	10	1	1	1	1	10	1	1	13	11	13	14	14	1	11	14	13	10	1	1	10	15
1	10	1	14	1	1	1	10	1	1	10	1	1	1	11	10	1	10	1	1	12	11	13	13	1	1	12	1
1	1	10	1	1	10	10	1	10	11	12	12	10	10	15	11	1	11	14	1	1	12	15	19	4	14	18	1
1	1	1	1	1	1	1	10	11	1	1	14	10	11	18	13	14	1	3	20	4	4	15	1	2	2	15	3
1	12	1	1	10	12	11	1	1	1	1	1	1	19	12	20	2	4	13	21	4	12	14	1	1	2	22	15
1	10	1	1	10	1	23	14	1	10	11	1	1	1	1	15	12	14	12	1	1	11	13	1	1	15	20	10
1	1	1	1	1	1	1	17	11	10	1	1	10	1	1	1	10	20	12	15	12	17	24	1	12	25	10	1
1	1	1	1	6	1	1	1	26	1	1	11	1	1	1	1	1	14	1	14	1	10	26	27	28	10	1	1
1	1	1	1	1	1	1	1	1	1	12	1	6	10	1	1	1	1	1	1	1	1	1	26	15	23	1	1
1	1	1	10	12	1	1	1	1	1	14	1	1	1	1	6	1	1	1	1	1	1	1	1	29	13	1	1
1	10	1	1	1	1	1	1	1	1	1	1	1	1	1	10	1	1	10	25	1	1	1	12	1	1	1	1
1	1	1	1	1	1	1	1	1	14	26	26	6	1	10	1	1	1	10	27	10	12	1	1	1	1	10	1
1	1	1	1	1	1	14	1	1	1	1	14	1	1	10	11	10	1	1	10	12	1	6	1	12	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	14	10	1	6	1	10	1	1	1	10
1	1	1	1	1	1	1	1	1	1	1	1	6	1	1	1	1	12	29	1	1	1	1	10	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	12	12	1	1	10	1	1	1	10	1	10	1	1	1	1	1	1
1	1	1	1	1	1	6	1	1	1	1	1	1	1	1	1	12	1	12	1	1	1	10	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	30	12	1	1	1	1	1	1	10	1	1
1	10	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	12	1	1	1	1	1	10

Symbol Graph Example



Size of Index vs. Frames



Experiments and Inter-class Similarity

Table 1: Before noise reduction inter-class similarity

Category 1	Category 2	Whole Frame Motion Representation Similarity	Symbolic Approach Similarity
run	walk	0,6644	0,4972
run	abrupt	0,6933	0,2001
walk	run	0,6644	0,4972
walk	abrupt	0,6738	0,3333
abrupt	run	0,6933	0,2001
abrupt	walk	0,6738	0,3333

Table 2: After noise reduction inter-class similarity

Category 1	Category 2	Whole Frame Motion Representation Similarity	Symbolic Approach Similarity
run	walk	0,4039	0,4148
run	abrupt	0,4510	0,0522
walk	run	0,2891	0,4148
walk	abrupt	0,5494	0,2073
abrupt	run	0,2829	0,0522
abrupt	walk	0,3709	0,2073

Noise Removal Effect

Table 1: Before noise reduction inter-class similarity

Category 1	Category 2	Whole Frame Motion Representation Similarity	Symbolic Approach Similarity
run	walk	0,6644	0,4972
run	abrupt	0,6933	0,2001
walk	run	0,6644	0,4972
walk	abrupt	0,6738	0,3333
abrupt	run	0,6933	0,2001
abrupt	walk	0,6738	0,3333

Table 2: After noise reduction inter-class similarity

Category 1	Category 2	Whole Frame Motion Representation Similarity	Symbolic Approach Similarity
run	walk	0,4039	0,4148
run	abrupt	0,4510	0,0522
walk	run	0,2891	0,4148
walk	abrupt	0,5494	0,2073
abrupt	run	0,2829	0,0522
abrupt	walk	0,3709	0,2073

Experiments (Semveillance dataset)

Behavior	Precision	Recall	F-measure
run	0.9656	0.7178	0.8231
walk	0.6741	0.9287	0.7746
abrupt	0.9522	0.9298	0.9408

Experiments (PETS04 dataset [Fisher, 2004])

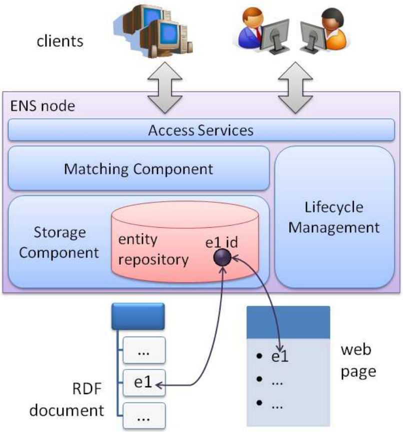
	Whole Frame Representation			With Frame Segmentation		
Behavior	Precision	Recall	F-measure	Precision	Recall	F-measure
browser	0.2093	0.3459	0.3273	0.8065	0.7366	0.7377
walker	0.9423	0.9491	0.9456	0.9918	0.8480	0.9129
fighters	0.1263	0.9461	0.2223	0.5608	0.8766	0.6437
meeters	0.2934	0.9810	0.4294	0.6685	0.8537	0.7448

Table 6: Experimental results for video indexing.

	Whole Frame Representation	With Frame Segmentation
Behavior	Specificity	Specificity
browser	0.3521	0.8444
walker	0.1829	0.8613
fighters	0.5257	0.8157
meeters	0.2605	0.8077

Outline

Entity Name System (ENS) [Bouquet et al., 2008, Palpanas et al., 2008]



An **ENS**:

- Maps real world entities to *Unique* identifiers.
- Provides for the *reuse* of identifiers.
- Supports *disambiguation* to real world entities in the Web.

Entity

Entity in the ENS is a set of:

- Free-form attribute names
- Free-form attribute values

Entity Example

```
title : Dr
firstName : Themis
family_name : Palpanas
homepage : http://dit.unitn.it/~themis
affiliation : University of Trento
```

Entity Subscription Services

An **Adaptive Entity Subscription System (AESS)** provides for:

- management of subscription to specific entities.
- the update of subscribers over changes to entities.
- informing subscribers over changes they are *mostly interested in*.
- takes into account explicitly or implicitly declared user interests.

Change Examples - Type and Content

Type: Deletion (of entity)
Content:(N/A)
or
Type: Entity Update, Attribute Update
Content: title→Prof
or
Type: Entity Update, Attribute Insertion
Content: affiliation→University of Trento

Our AESS

- expects user feedback for interest indication.
- expresses interest as a real value.
- defines predefined values for interest levels.

We need to

- create an architecture for the system.

Our AESS

- expects user feedback for interest indication.
- expresses interest as a real value.
- defines predefined values for interest levels.

We need to

- create an architecture for the system.
- represent efficiently the *type* and *content (i.e. free form strings)* info of a change.

Our AESS

- expects user feedback for interest indication.
- expresses interest as a real value.
- defines predefined values for interest levels.

We need to

- create an architecture for the system.
- represent efficiently the *type* and *content (i.e. free form strings)* info of a change.
- create a user model, from user feedback, that can use this representation.

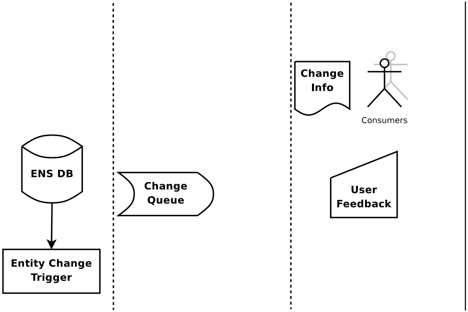
Our AESS

- expects user feedback for interest indication.
- expresses interest as a real value.
- defines predefined values for interest levels.

We need to

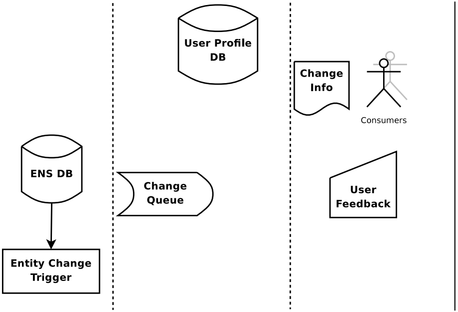
- create an architecture for the system.
- represent efficiently the *type* and *content (i.e. free form strings)* info of a change.
- create a user model, from user feedback, that can use this representation.
- take into account both simple and complex scenarios of preference.

Architecture



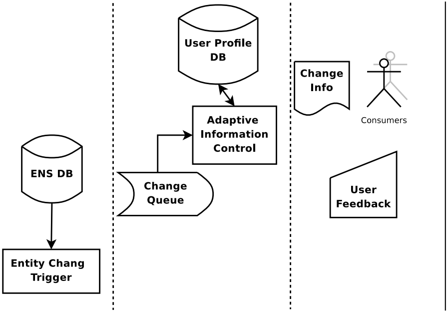
- Change Queue

Architecture



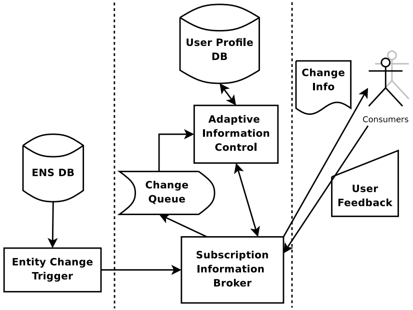
- Change Queue
- User Profile DB

Architecture



- Change Queue
- User Profile DB
- Adaptive Information Control

Architecture



- Change Queue
- User Profile DB
- Adaptive Information Control
- Subscription Information Broker

Outline

Representing Changes — Type

The **type of a change**:

- deletion, splitting, merging or update
- updates can involve: attribute deletion, attribute insertion or attribute update
- given graded indication of normality of the change, e.g. 0 (abnormal) to 1 (normal)

Feature Space

A dimension indicative of each type/subtype of change.

But what about *Content*?

Representing Changes — Content

The **content of a change**:

- Instances of attribute names
- Instances of attribute values

Problems and requirements

- Free form strings
- Other types (numeric, date, etc.)
- Fuzzy string matching
- Updatable model — if possible
- Graded similarity from comparison of instance to model

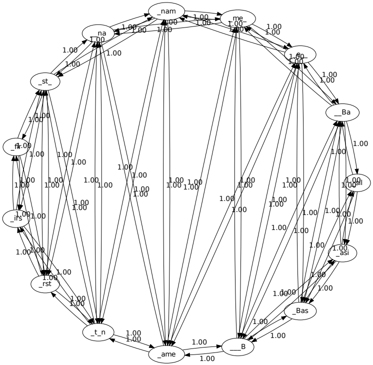
We use **Character N-gram Graphs**

Content as a graph

A character n-gram graph is a string model based on the coexistence of character n-grams in a string.

first_name:Basil

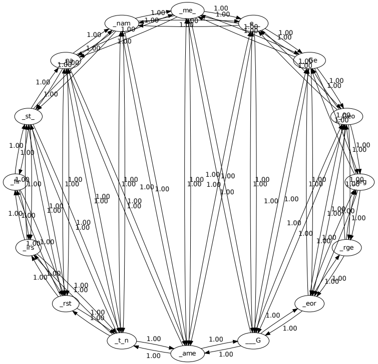
Graph Size: 39 bidirectional edges



Content as a graph: Updating

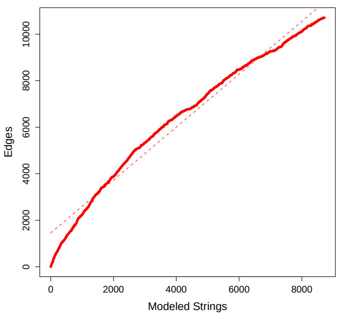
first_name:Basil
first_name:George

Graph Size: 42 bidirectional edges

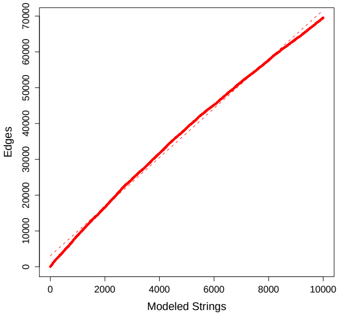


i.e., not bad scaling for normal user requirements.

Text Size to Graph Size — Actual vs Random



Actual Text



Random Text

Mapping Content to the Feature Space

Given
a set of labeled changes and *a new change*.
We want
dimensions indicative of content similarity.

N-gram Graph Normalized Value Similarity (NVS)

- Create a graph representing labeled instances for each level.
- We have one similarity-based feature for each interest level.

Outline

From Change to Feature Space: Workflow Overview

To map a change instance to the feature space:

- Apply values to *Type* dimensions.
- Calculate *Content* graph similarities for every interest level.
- Add dimensions for *Content* graph similarity.

Update Model with New Data

To update the user model with a new instance:

- Merge *Content*-based graph into corresponding interest level graph.
- Initialize new vector.
- Calculate an ϵ-SVR [Chang and Lin, 2001, Vapnik, 1998] regression model to estimate interest.

How does this model perform?

Outline

Experimental Setting

- Synthetic data for data changes
- 10-fold validation
- 1000 changes per fold
- Each iteration is mapped to a set of 10 changes

We judge

- if learning occurs and its rate.
- if the use of content (graphs) is useful.

Determining Learning Ability

Table: Correlation between Emission-Iteration Number and Regression Mean Absolute Error per Subscriber Profile and Method

Subscriber	*Graphs*	*Correlation (p-value)*
Type-based	✓	**-0.3398559** ($< 10^{-2}$)
		-0.2993715 ($< 10^{-2}$)
Attribute name-based	✓	**-0.3734062** ($< 10^{-2}$)
		-0.03564642 (0.2601)
Attribute name-value pair-based	✓	**-0.08581718** ($< 10^{-2}$)
		-0.02968072 (0.3484)
Complex	✓	**-0.5989662** ($< 10^{-2}$)
		-0.03393356 (0.2837)

Determining Speed and Stability of Learning (1)

Rate of Acceptable Errors (RAE): Percentage of errors in estimation that cannot cause ranking error.

Black line: Content aware method. Gray line: Simple method.

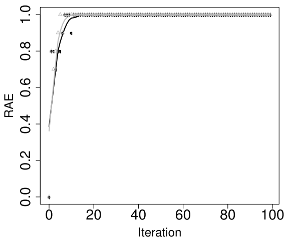


Figure: Type Based

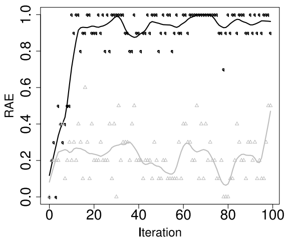


Figure: Attribute Name-based

Determining Speed and Stability of Learning (2)

Rate of Acceptable Errors (RAE): Percentage of errors in estimation that cannot cause ranking error.

Black line: Content aware method. Gray line: Simple method.

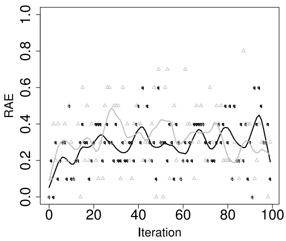


Figure: Attribute Name-Value-based

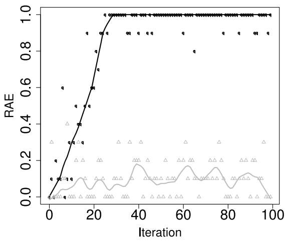


Figure: Complex

Importance of Representation

Rate of Acceptable Errors (RAE): Percentage of errors in estimation that cannot cause ranking error.

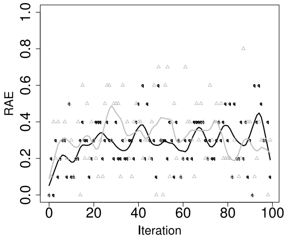


Figure: Name, Value in One Graph

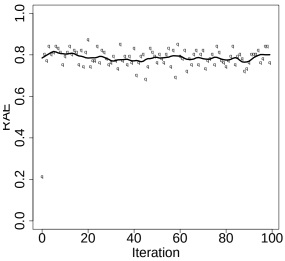


Figure: Name, Value in Separate Graphs

Outline

Application suite

- AutoSummENG (plus new version)
- MUDOS-NG
- Document Classifier
- Spam filter
- Grammaticality Estimator
- Entropy-based Chunk Splitter

Library

- Character and Word N-gram Graphs
- N-Gram Distribution Graphs
- Operators
- **Serializability**
- Distributed Processing Examples (JADE)
- Multi-threading
- Utilities (file to string, Distribution class, etc.)
- Interoperability (R, thesauri, etc.)

Outline

Open Source

- LGPL
- Extendable
- Reusable
- Lots of examples
- Non-trivial implementations

Easy to Apply

- Find what the vertices should be
- Define the neighborhood relation
- Use them

Outline

Almost there...

Flashback

- Optical Flow Proximity Graphs
- Proximity Graphs in a Hierarchy
- Combining Graphs with Vector Space
- JINSECT Toolkit and Library

Into the future...

- Indexing graphs
- Hierarchy and granularity criteria
- Expressiveness of proximity graph
- Recognition of n-gram graphs

Into the future...

- Indexing graphs
- Hierarchy and granularity criteria
- Expressiveness of proximity graph
- Recognition of n-gram graphs
- ...and whatever **you** plan to make out of them.

N-gram Graphs: A New Perspective

Thank you

George Giannakopoulos (ggianna@disi.unitn.it)

Please provide your thoughts on the feedback form[1].

[1] See `http://tinyurl.com/2fna572`

Outline

Change Data Generation: User Simulation

User type (Prob.)	*Change type*	*Probability*
Benevolent (0.95)	Attribute change (normal)	0.60
	Attribute insertion	0.30
	Attribute deletion	0.10
Sys.admin.(0.03)	Entity merge	0.45
	Entity split	0.45
	Entity deletion	0.10
Malevolent (0.02)	Attribute change (abnormal)	0.70
	Attribute deletion	0.30

Subscription User Simulation

Subscriber	*Importance*	*Description*
Type-based	Critical	Attribute deletion.
	Interesting	Entity deletion.
Attribute name-based	Critical	Any change concerning an attribute that contains the string "name".
	Interesting	(None)
Attribute name-value pair-based	Critical	Attribute change or insertion on "isDeceased" attribute, with a new value of "true".
	Interesting	Attribute change or insertion on "isDeceased" attribute, with a new value of "false".
Complex	Critical	Default attribute (some attributes in the ENS are considered default — e.g., the name of a person entity — while all the others non-default) update or insertion with an abnormal value.
	Interesting	Default attribute deletion or normal update.

Bouquet, P., Stoermer, H., and Bazzanella, B. (2008).
An entity name system (ENS) for the semantic web.
In *ESWC*, pages 258–272.

Chang, C.-C. and Lin, C.-J. (2001).
LIBSVM: a library for support vector machines.
Software available at
http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Fisher, R. (2004).
The PETS04 surveillance ground-truth data sets.
In *Proc. 6th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, pages 1–5.

Palpanas, T., Chaudhry, J. A., Andritsos, P., and Velegrakis, Y. (2008).
Entity data management in OKKAM.
In *DEXA Workshops*, pages 729–733.