

# Natural Language Generation: Traditional Approaches and Research Directions

## Lecture 1: How we got here

Pablo Ariel Duboue, PhD.

Textualization Software Ltd.  
Vancouver, Canada

17th Estonian Summer School on Computer and Systems Science

# Outline

## 1 Natural Language Generation

- These Lectures
- The NLG Problem

## 2 NLG Methods

- NLG Architecture
- NLG Subtasks
- Linguistic Theories

## 3 Examples

- Domains
- Tools
- Thoughtland

# Outline

## 1 Natural Language Generation

- These Lectures
- The NLG Problem

## 2 NLG Methods

- NLG Architecture
- NLG Subtasks
- Linguistic Theories

## 3 Examples

- Domains
- Tools
- Thoughtland

# Lecture 1

- We will cover **traditional approaches** used early on in the field:
  - knowledge-based, and
  - linguistic.
- This will help us:
  - understand the **study questions of the field**, and
  - put the progress made using machine learning methods in perspective.
- Thoughtland n-dimensional Object Description System.

## Lecture 2

- We will discuss **statistical approaches** to NLG,
  - with a focus on **data acquisition**
  - and **evaluation methods**.
- ProGenIE Biography Generation System.

## Lecture 3

- We will discuss the main architectures proposed for using **artificial neural networks** for natural language generation problems.
- We will start with a review of Deep Learning concepts.
- Keywords4bytecodes reverse engineering of Java bytecodes project.

# About the Presenter

- Columbia University in the city of New York.
  - Doctoral Dissertation: “Indirect Supervised Learning of Strategic Generation Logic”, defended Jan. 2005.
- IBM Research Watson.
  - Deep QA - Watson - Jeopardy! Show.
- In Montreal from 2010-2016.
  - Doing consulting for startups.
  - Two semesters teaching at Cordoba University (NLG / ML on large datasets).
- In Vancouver since 2017.
  - Started Textualization Software Ltd.
    - Mission: on NLG for Data Science.

# About the Audience

- How many people have Natural Language Processing background?
  - NLG?
- How many people have Machine Learning background?
  - Deep Learning?
- How many people have taken a Compilers class?
  - Reverse Engineering?
- These lectures **main take away**:
  - Interesting problem, focus on **creation** by computer.
    - Learn some NLP from a different perspective.
  - Generate-and-Rank approach (second lecture).
  - Encoder-Decoder framework (third lecture).



# Outline

## 1 Natural Language Generation

- These Lectures
- The NLG Problem

## 2 NLG Methods

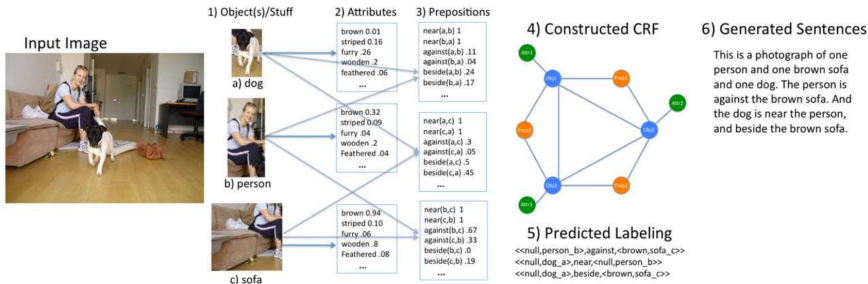
- NLG Architecture
- NLG Subtasks
- Linguistic Theories

## 3 Examples

- Domains
- Tools
- Thoughtland

# Full End-to-End NLG Example

- Kulkarni et al. 2011, adapted from Bernardi et al. 2017



## Study Questions for NLG

- How to choose among equally valid options? (Example from Paiva & Evans, 2005)
  - *The dose of the patient's medicine is taken twice a day. It is two grams.*
  - *The two-gram dose of the patient's medicine is taken twice a day.*
  - *The patient takes the two-gram dose of the patient's medicine twice a day.*
- How to represent such decisions?
- How to revise the decisions if they lead through the wrong path?
- Other issues:
  - Representing intention.
  - Psycholinguistically plausible incremental generation.

# Input to the NLG system

- Where the data comes from?
  - The existence of data to generate from is part of the problem definition.
  - The data is available and should not be changed to better accommodate the NLG system.
  - A potential source of data is the output of Information Extraction system, for example look at <http://ie4opendata.org> another project by the same presenter.
- Common inputs:
  - Purely tabular
  - Frames, encoded in S-Expression (trees).
  - Semantic graphs, encoded in RDF (graphs).
  - Mixed language and data.

# Input: Tabular

- Excel spreadsheets, for example:

	A	B	C	D	E	F	
1	Item	Category	Price	Profit	Actual Profit	Calories	
2	Beer	Beverages	\$ 4.00	50%	\$ 2.00	200	
3	Hamburger	Hot Food	\$ 3.00	67%	\$ 2.00	320	
4	Popcorn	Hot Food	\$ 5.00	80%	\$ 4.00	500	
5	Pizza	Hot Food	\$ 2.00	25%	\$ 0.50	480	
6	Bottled Water	Beverages	\$ 3.00	83%	\$ 2.50	0	
7	Hot Dog	Hot Food	\$ 1.50	67%	\$ 1.00	265	
8	Chocolate Dipped Cone	Frozen Treats	\$ 3.00	50%	\$ 1.50	300	
9	Soda	Beverages	\$ 2.50	80%	\$ 2.00	120	
10	Chocolate Bar	Candy	\$ 2.00	75%	\$ 1.50	255	
11	Hamburger	Hot Food	\$ 3.00	67%	\$ 2.00	320	
12	Beer	Beverages	\$ 4.00	50%	\$ 2.00	200	
13	Hot Dog	Hot Food	\$ 1.50	67%	\$ 1.00	265	
14	Licorice Rope	Candy	\$ 2.00	50%	\$ 1.00	280	
15	Chocolate Dipped Cone	Frozen Treats	\$ 3.00	50%	\$ 1.50	300	
16	Nachos	Hot Food	\$ 3.00	50%	\$ 1.50	560	
17	Pizza	Hot Food	\$ 2.00	25%	\$ 0.50	480	
18	Beer	Beverages	\$ 4.00	50%	\$ 2.00	200	

# Input: Frames

- Hierarchical Records of Attribute-Value pairs.
  - Each one has a unique name and a type.
  - Possibility of an ontology describing which attributes are possible for each type.

```
name-first (person -1, 'Sean')
name-last (person -1, 'Connery')
occupation (person -1, c-actor)
occupation (person -1, c-producer)
ex-spouse (person -1, person -2)
spouse (person -1, person -4)
name-first (person -2, 'Diane')
name-last (person -2, 'Cilento')
occupation (person -2, c-actress)
name-first (person -4, 'Micheline')
name-last (person -4, 'Roquebrune')
relative (person -1, c-son, person -3)
occupation (person -4, c-painter)
relative (person -2, c-son, person -3)
```

```
name-first (person -3, 'Jason')
name-last (person -3, 'Connery')
work (person -1, bond-1)
title (bond-1, 'James_Bond')
work (person -1, movie-2)
title (movie-2, 'operation_warhead')
award (person -1, oscar-1)
title (oscar-1, 'Oscar')
sub-title (oscar-1, 'Best_Actor')
reason (oscar-1, bond-1)
eye-color (person -1, c-green)
accent (person -1, c-scottish)
```

# Input: RDF

- Semantic Web format for frames.

```
<rdf:Description about="http://www.dlib.org/dlib/may98/05contents.html">
  <dc:Title>DLIB Magazine – The Magazine for Digital Library Research
    – May 1998</dc:Title>
  <dc:Description>D-LIB magazine is a monthly compilation of
    contributed stories, commentary, and briefings.</dc:Description>
  <dc:Contributor rdf:parseType="Resource">
    <dcq:AgentType
      rdf:resource="http://purl.org/metadata/dublin_core_qualifiers#Editor"/>
    <rdf:value>Amy Friedlander</rdf:value>
  </dc:Contributor>
  <dc:Publisher>Corporation for National Research Initiatives</dc:Publisher>
  <dc:Date>1998-01-05</dc:Date>
  <dc:Type>electronic journal</dc:Type>
  <dc:Subject>
    <rdf:Bag>
      <rdf:li>library use studies</rdf:li>
      <rdf:li>magazines and newspapers</rdf:li>
    </rdf:Bag>
  </dc:Subject>
  <dc:Format>text/html</dc:Format>
  <dc:Identifier>urn:issn:1082-9873</dc:Identifier>
  <dc:Relation rdf:parseType="Resource">
    <dcq:RelationType
      rdf:resource="http://purl.org/metadata/dublin_core_qualifiers#IsPartOf"/>
    <rdf:value resource="http://www.dlib.org"/>
  </dc:Relation>
```

# RDF Triples: DBpedia and Wikidata










- Large source of frame data in RDF format:

```

{{About|the 43rd President of the United States|his father, the 41st President|George
H. W. Bush|the American settler|George Washington Bush}}
<!-- See [[WP:EDN]] -->
{{Pp-move-indef}}
{{Active editnotice}}
{{Use mdy dates|date=November 2016}}
{{Infobox president
|name           = George W. Bush
|office         = [[List of Presidents of the United States|43rd President of the
United States]]
|image          = George-W-Bush.jpeg
|predecessor    = [[Bill Clinton]]
|successor      = [[Barack Obama]]
|vicepresident  = [[Dick Cheney]]
|order2         = [[List of Governors of Texas|46th Governor of Texas]]
|lieutenant2    = {{ublist|[[Bob Bullock]]|Rick Perry}}
|predecessor2   = [[Ann Richards]]
|successor2     = [[Rick Perry]]
|birth_name     = George Walker Bush
|birth_date     = {{Birth date and age|1946|7|6}}
|birth_place    = {{nowrap|[[New Haven, Connecticut]], U.S.}}
|party          = [[Republican Party (United States)|Republican]]
|spouse         = {{Marriage|[[Laura Bush|Laura Welch]]|November 5, 1977}}
|relations      = ''See [[Bush family]]''
|children       = {{Hlist|[[Barbara Bush (born 1961)|Barbara]]|[[Jenna Bush

```

dbpedia.org/page/George\_W.\_Bush

 Browse using       Formats  

crisis.Bush left office in 2009, and f  
a home in a suburban area of Dalla  
written a memoir, Decision Points. f  
(en)

dbo:almaMater

- dbr:Yale\_College
- dbr:Harvard\_Business\_School

dbo:award

- dbr:United\_States\_Aviator\_Badge
- dbr:National\_Defense\_Service\_Med
- dbr:Air\_Force\_Outstanding\_Unit\_Aw
- dbr:Marksmanship\_Ribbon

dbo:birthDate

- 1946-07-06 (xsd:date)



# Mixed Input

- Text-to-text generation.
  - Not covered in these lectures.
    - Covered by other areas of NLP but not NLG.

# Input: Role of Intention

- Intention or Goal is the purpose for which the generated text or speech will be used.
  - It is part of the input to the NLG system.
    - Even though many times is left implicit.
- Most NLG systems seeks to **inform** the user about certain data.
  - But many uses of language have an intention besides just making the hearer aware of certain data.
- We might want to...
  - ...convince them about the truth value of certain piece of data.
  - ...get them to act on a given topic.
  - ...stop performing certain actions.
  - ...help them retain the given information

# Input: Role of User Model

- Last input item to an NLG system: **the user model**.
  - Again, many times left implicit
- It is very important to have an idea of the person we are communicating with:
  - To shape the content in a way that helps them understand it.
  - Avoid saying things that are already known to them.
- “Know your audience” is a well held advice for writers and presenters alike

# NLP and NLG

- Natural Language Processing (NLP) is further divided into:
  - Natural Language Understanding (NLU).
  - Natural Language Generation (NLG).
- NLU pipeline:
  - Tokenization.
  - Part-of-Speech tagging.
  - Chunking.
  - Parsing.
  - Semantic parsing (arguments).
  - Discourse parsing.
  - Pragmatics.

# Is NLG the Inverse of NLU?

- In general terms, the answer is no.
- For particular subtasks, like surface realization, there is a NLU counterpart (such as parsing).
- However, NLU seldom builds from the text structures rich and complex enough to allow for generation.
  - It will build a structure that allows for generation of the original sentence, and many others
  - Without enough information to choose that particular sentence
  - That is not a minor issue, as NLG central issue is choice.
- This problem is more complicated when dealing with information not present in the text (such as in Content Selection).
  - Did the author ignore the fact or choose not to include it in the text?

# Automatic Text Construction without NLG

- Plenty of text constructed automatically by computers.
  - Most of such texts are not built using NLG techniques.
- How programmers solve this problem without using NLG?
  - Templates "pay \$payee \$amount of dollars"
  - If statements, lots of ifs:
    - `if amount == 1: "pay $payee $amount dollar"`
  - The feeling that there must be a better way.
- NLG is the better way:
  - A sensible way to encode those piles of `if` statements.
  - NLG at its core is about encoding and handling choices.
    - Iterative enrichment of underspecified structures.

# NLG Community

- Venues:
  - INLG.
  - ACL.
- Companies:
  - ARRIA NLG.
  - Narrative Science.
  - Automated Insights.
- Groups:
  - CLAN at Aberdeen in Scotland, UK.
  - SyNaLP at LORIA in Nancy, France.
  - Applied Data Analytics Research and Enterprise Group in Brighton, UK.
  - Institute of Linguistics and Language Technology, Malta.
  - Columbia NLP group, New York, US.
  - Institute for Language, Cognition and Computation, Edinburgh, UK.
  - MIT CSAIL, Boston, US.

# Outline

- 1 Natural Language Generation
  - These Lectures
  - The NLG Problem
- 2 NLG Methods
  - NLG Architecture
  - NLG Subtasks
  - Linguistic Theories
- 3 Examples
  - Domains
  - Tools
  - Thoughtland



# NLG Architecture

- Pipeline or blackboard?
  - In a pipeline subtasks are executed in order, with the their output becoming the input of the next task.
  - In a blackboard, a common structure (for example, a DB) contains shared information from tasks.
    - Tasks can be executed in any order and even re-executed as more information becomes available.
- Which subtasks (and in which order)?
- The classic architecture is a pipeline with two stages:
  - **Strategic Generation**: What to say.
  - **Tactical Generation**: How to say it.

# NLG Architecture

- A common strategy divides Tactical Generation into two subtasks:
  - **Sentence Planning.**
    - How to arrange the information into sentences.
  - **Surface Realization.**
    - How to take the information that fits into a sentence into an actual sentence.
    - Many times when people say that NLG is the inverse of NLU, they think of reversible surface realization / parsing.

# NLG Architecture

- Plenty of work going away from pipeline and the three stages.
  - In lower stages of the system we might realize earlier decisions were wrong and need to be revisited.
- Evidence that a revision based system it is to be preferred.
  - Not that different from how humans author high quality text.
- How about the way humans speak?
  - Sometimes incorrect speech is produced and we have repairs.
  - A special architecture is needed to generate such behaviour.
    - And work has been done on that front.
    - It is unclear whether we want stuttering computers at all.

# Outline

- 1 Natural Language Generation
  - These Lectures
  - The NLG Problem
- 2 NLG Methods
  - NLG Architecture
  - NLG Subtasks
  - Linguistic Theories
- 3 Examples
  - Domains
  - Tools
  - Thoughtland

# Subtasks

- Content Planning
  - Content Selection
  - Document Structuring
- Sentence Planning
  - Aggregation
  - Referring Expression Generation
  - Lexicalization
- Surface realization
  - Linearization

# Content Planning

- Input: Raw Data
- Output: Document Plan
- Subtasks:
  - Content Selection
  - Document Structuring
- Example:

```
[intro-person(person-1),]
[ex-spouse(person-1,person-2), intro-person(person-2),
 spouse(person-1,person-3), intro-person(person-3),]
[child(person-1,person-4), intro-person(person-4),]
[movie(bond-1,person-1), intro-award(oscar-1,person-1)]
```

# Content Selection

- Decides what information will appear in the output text
  - This depends on
    - intention
    - audience
    - available input
    - other constraints such as allowed text length

# Document Structuring

- Decides how chunks of content should be grouped in a document
  - how to relate these groups to each other
  - in what order they should appear.
- For example, when describing last month's weather, you might talk first about temperature, then rainfall.
  - Or you might start off generally talking about the weather and then provide specific weather events that occurred during the month.



# Sentence Planning

- Input: Document Plans
- Output: Sentence Plans
- Subtasks:
  - Merging messages to make complex sentences: Aggregation
  - Giving words to concepts: Lexicalization and Referring  
Expression Generation

# Aggregation

- Decides how the structures created by document planning should be mapped onto linguistic structures
  - such as sentences and paragraphs.
- For example, two ideas can be expressed in two sentences or in one:
  - *The month was cooler than average. The month was drier than average.*
  - *The month was cooler and drier than average.*

# Referring Expression Generation

- Decides which expressions should be used to refer to entities (both concrete and abstract).
  - The same entity can be referred to in many ways.
  - For example, March of last year can be referred to as:
    - *March 2017*
    - *March*
    - *March of the previous year*
    - *it*

# Lexicalization

- Decides what specific words should be used to express the content.
- Actual nouns, verbs, adjectives and adverbs to appear in the text are chosen from a lexicon.
  - Particular syntactic structures are chosen as well.
  - For example
    - *'the car owned by Mary'*
    - *'Mary's car'*

# Surface Realization

- Input: Sentence Plans
- Output: Strings
- Uses rules of grammar (about morphology and syntax) to convert abstract representations of sentences into actual text.
  - Operates over a tree
- Structure realization converts abstract structures such as paragraphs and sentences into mark-up symbols which are used to display the text.
- Subtask: Linearization

# Linearization

- Once the tree is fully specified, it needs to be linearized to obtain a final string
- Also need to be made orthographical modifications (like capitalization of first word, etc)

# Outline

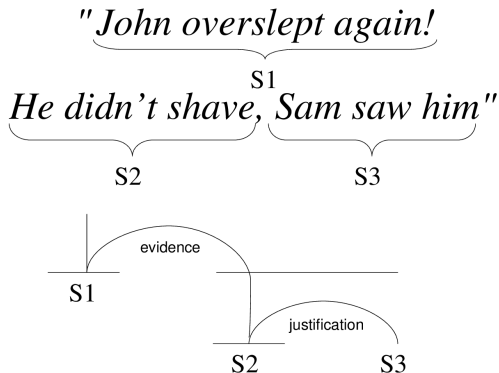
- 1 Natural Language Generation
  - These Lectures
  - The NLG Problem
- 2 NLG Methods
  - NLG Architecture
  - NLG Subtasks
  - Linguistic Theories
- 3 Examples
  - Domains
  - Tools
  - Thoughtland

# Linguistic Theories Employed

- Some linguistic theories employed in NLG:
  - Meaning-to-Text.
  - Functional Unification Grammars.
  - Systemic Functional Linguistics.
  - **Rhetorical Structure Theory.**
  - Discourse Rhetorical Theory.
- While NLG is at the “application” end of theories, sometimes it helps also as validation discipline:
  - Centering Theory and pronominalization.



# Rhetorical Structure Theory



# Outline

- 1 Natural Language Generation
  - These Lectures
  - The NLG Problem
- 2 NLG Methods
  - NLG Architecture
  - NLG Subtasks
  - Linguistic Theories
- 3 **Examples**
  - **Domains**
  - Tools
  - Thoughtland

# Domains for NLG

- Weather
- Stocks
- Sports
- Tourism
- Medical
- Sensor data

# Weather

- Weather Reports are the **quintessential** NLG application.
- Particularly for multilingual environments or environments that require customized weather reports consumed by a small group of people.
  - Maritime weather reports.
  - Example (from Reiter & Sripada, 2002):

*FORECAST 00-24 GMT, WEDNESDAY, 04-Oct 2000  
WIND(10M): WSW 20-24 BACKING SSW 10-14 BY MIDDAY  
THEN  
VEERING SW 24-28 BY EVENING*

# Stocks

- ANA (from Kukich, 1983):

*Thursday June 24, 1982*

*wall street's securities markets meandered upward through most of the morning, before being pushed downhill late in the day yesterday. the stock market closed out the day with a small loss and turned in a mixed showing in moderate trading.*

*the Dow Jones average of 30 industrials declined slightly, finishing the day at 810.41, o 2.76 points. the transportation and utility indicators edged higher.*

*volume on the big board was 558,600,000 shares compared with 627,100,000 shares on Wednesday. advances were ahead by about 8 to 7 at the final bell.*

- More recent work by Schilder and others (Thompson Reuters)  
 see Smiley et al, 2016.

# Sports

- Streak (Robin, 1994), revision-based generation:

*Draft 0: Dallas, TX -- Charles Barkley registered 42 points Friday night as the Phoenix Suns routed the Dallas Mavericks 123 - 97.*

*Draft 5: Dallas, TX -- Charles Barkley matched his season record with 42 points and Danny Ainge came off the bench to add 21 Friday night as the Phoenix Suns handed the Dallas Mavericks their franchise worst 27th defeat in a row at home 123 - 97.*

- More recent work, summarizing Robocup soccer games by Mooney and others, see Chen & Mooney, 2008.

# Sensor Data

- Gas Turbines, SumTime-Turbine (Yu et al, 2005):

*Vertical aggregation (Linguistic format)*

*Spikes in all channels at 21:49:49, 21:51:31, 21:53:17, 23:55:57*

*Mostly spikes with some steps in all channels at 22:35:02*

*Horizontal aggregation (Linguistic format)*

*Spikes in all channels at 21:49:49 and 23:55:57*

*Mostly spikes with some steps in all channels at 22:35:02,*

# Tourism

- Museums, ILEX system, see Mellish et. al, 1998.

This jewel is made from diamonds, yellow metal, pearls, oxidized white metal and opals.

It was made in 1976 and was made in London.

This jewel draws on natural themes for inspiration: it uses natural pearls.

It was made by Flockinger who is an English designer.

Flockinger lived in London which is a city.

This jewel is a necklace and is set with jewels.

It is encrusted with jewels and is an Organic style jewel: it has silver links encrusted asymmetrically with pearls and diamonds.

Indeed, Organic style jewels are usually encrusted with jewels.

Organic style jewels usually draw on natural themes for inspiration and are made up of asymmetrical shapes.

Organic style jewels usually have a coarse texture.

This jewel is 72.0 cm long.

The previous jewel has little diamonds scattered around its edges and has an encrusted bezel. It is encrusted with jewels: it features diamonds encrusted on a natural shell.

- More recent work, see Stock et al, 2007.



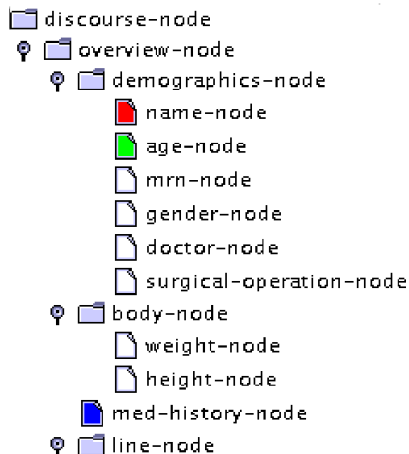
# Medical

- MAGIC (next).
- Neonatal care, see Gatt et al, 2009.
- Smoke Cessation letters, see Reiter et al, 2003.

# MAGIC

- Intelligent multimedia presentation system.
- Medical domain:
  - Reporting cardiac surgery patient status.
  - Time critical.

# MAGIC: Content Plans



“**J. Doe** is a **seventy-eight year-old** male patient of Doctor Smith undergoing aortic valve replacement. His medical history includes **allergy to penicillin and congestive heart failure**. He is sixty-six kilograms and one hundred sixty centimeters. The patient is thirty minutes post bypass and will be arriving in the unit shortly. His infusion lines include a Swan Ganz in the right IJ and an IV in each arm.

# MAGIC: Aggregation

- MAGIC was a test-bed for linguistic-based aggregation research that culminated in Shaw, 2002 thesis:

The patient is John Doe.

He is 71 year-old.

He is male.

The patient's surgeon is Jane Doe.

He undergoes tricuspid valve replacement and coronary artery bypass grafting.

His weight is 64 kilograms.

His height is 172 centimeters.

His infusion lines include a Swan Ganz in the right IJ

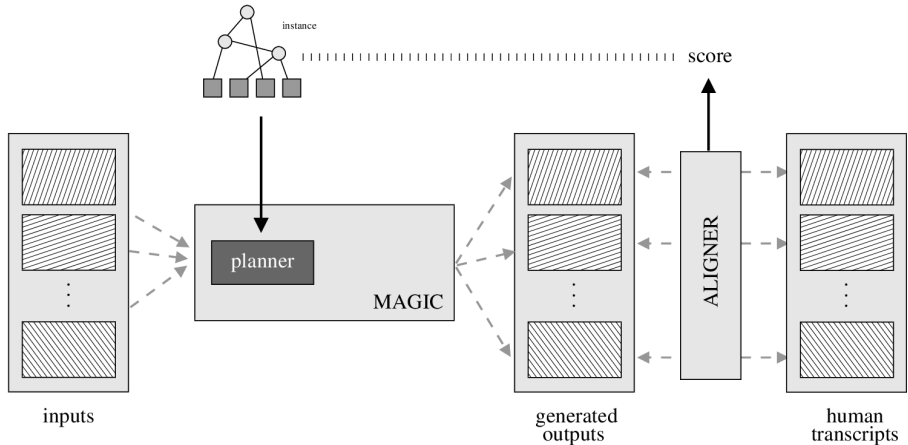
John doe is a 71 year-old male patient undergoing tricuspid valve replacement and coronary bypass.

The patient's surgeon is Jane Doe.

His weight is 64 kilograms and his height 172 centimeters.

His infusion lines include a Swan Ganz in the right IJ, an arterial line in the right radial artery and an IV in his right arm.

# MAGIC: My Work



# MAGIC: Other Work

- Multimedia generation.
  - Placement of 3D labels.
- Speech generation (prosody / intonation).
- Information extraction for medical history.

# Outline

## 1 Natural Language Generation

- These Lectures
- The NLG Problem

## 2 NLG Methods

- NLG Architecture
- NLG Subtasks
- Linguistic Theories

## 3 Examples

- Domains
- **Tools**
- Thoughtland

# Example Tools

- I will present now some tools to do NLG subtasks and end-to-end systems that are:
  - Free Software (open source).
  - I have contributed myself or I am familiar with them.
  - Relatively recent.
- These tools are useful if want to:
  - Use these techniques for your own projects.
  - Learn more by taking a detail view under the hood.
- I am always looking for new collaborators for Free Software projects:
  - <http://duboue.net/collaboration.html>



# A Java Surface realizer

- Most surface realizers require a complex sentence plan:
  - Usually tied to a complex linguistic formalism.
  - Requiring users well versed with said formalism.
- They also make it very difficult (if not impossible) to mix sentence plans with raw text.
  - Mixed-level input.
- SimpleNLG is a straightforward surface realizer intended for use by programmers rather than NLG experts:
  - It provides much less functionality than other realizers.
  - It is very easy to use.

# SimpleNLG Tutorial

```
Lexicon lexicon = Lexicon.getDefaultLexicon();  
NLGFactory nlgFactory = new NLGFactory lexicon);  
Realiser realiser = new Realiser lexicon);  
NLGElement s1 = nlgFactory.createSentence("my dog is  
happy");  
String output = realiser.realiseSentence(s1);  
System.out.println(output);
```

- My dog is happy.

# SimpleNLG Tutorial

```
SPhraseSpec p = nlgFactory.createClause();  
p.setSubject("Mary");  
p.setVerb("chase");  
p.setObject("the monkey");  
String output = realiser.realiseSentence(p);  
System.out.println(output);
```

- \*Mary chase the monkey.

# SimpleNLG Tutorial

```
NPPhraseSpec subject1 =  
nlgFactory.createNounPhrase("Mary");  
NPPhraseSpec subject2 =  
nlgFactory.createNounPhrase("your", "giraffe");  
CoordinatedPhraseElement subj =  
nlgFactory.createdCoordinatedPhrase(subject1,  
subject2);  
p.setSubject(subj);
```

- Mary and your giraffe chase the monkey.

# SimpleNLG Tutorial

```
NPPhraseSpec object1 =  
nlgFactory.createNounPhrase("the monkey");  
NPPhraseSpec object2 =  
nlgFactory.createNounPhrase("George");  
CoordinatedPhraseElement obj =  
nlgFactory.createdCoordinatedPhrase(object1,  
object2);  
obj.addCoordinate("Martha"); p.setObject(obj);  
obj.setFeature(Feature.CONJUNCTION, "or");
```

- Mary and your giraffe chase the monkey, George or Martha.

# SimpleNLG Tutorial

```
p.setFeature(Feature.TENSE, Tense.FUTURE);  
p.setFeature(Feature.NEGATED, true);
```

- Mary will not chase the monkey.

# Some Interesting Applications of SimpleNLG

- Thoughtland: describing n-dimensional objects (end of today's lecture)
- An icon-based speech communicator for Disabled Children - Android App - Alternative communication (AAC)
  - <https://github.com/vidma/aac-speech-android>

# Android App - Alternative communication (AAC)



**verbs**

annoying	celebrate	to eat
to like	I need help	I want
I want that one	Noise is disturbing me	annoying



ME	NOUS	PASSÉ	VERBES	FUTUR
TE	VOUS	NÉGATION	ÊTRE	VOULOIR
LUI	EUX	AVOIR	POUVOIR	QUESTION
LUI	EUX	QUALITÉS	ALIMENTS	OBJETS
ON	ÇA	ADV.PLACE	EXPR.COMUN	POINT



## Other Tools: OpenSchema

- Document Structuring Schemata are strategic generation tool.
  - Starting from a pool of coarsely relevant knowledge (the relevant knowledge pool), schemata are **finite state machines** with terminals in a language of rhetorical predicates.
- They were invented by Dr. McKeown for her thesis work.
  - Analyzed a texts of expository in nature in different genres.
    - found out four different organizational strategies that captured a large number of texts.
  - These four strategies are “schemata” and defined as

(...) a representation of a **standard pattern of discourse structure** which efficiently encodes a set of communicative techniques that a speaker can use for a particular discourse purpose.

(McKeown, 1985, p.20)

## Other Tools: OpenSchema

- OpenSchema extends McKeown's work by providing a declarative definition of her predicates,
- now not necessarily rhetorical in nature
  - communicative predicates)
- A Java library that implements schemata
  - Written on its own DSL (domain specific language)
- <http://openschema.sourceforge.net/>

# OpenSchema: Example

```
predicate attributive
  variables
    req object : c-entity
    def attribute : c-entity
  properties
    attribute == object.has-a
  output
    template "{{object-name}} has a {{attribute-name}}".
    object-name object.name
    attribute-name attribute.name
    object object
    attribute attribute

schema attributive(object: c-entity)
  attributive(entity|object)
  amplification(entity|object)
  star
    particular-illustration(entity|object, attribute|attribute)
  optional
    classification(entity|object, attribute|attribute)
  analogy(entity|object, analogy-pred|analogy)
  choice
    explanation(entity|object, issue|analogy)
```

# OpenSchema: Biography Example

```

predicate pred-person
  variables
    req def person : c-person
    occupation : c-occupation
  properties ; properties that the variables have to hold
    occupation == person.occupation
  output
    template "{{name-first}} {{name-last}} is a {{occupation}}".
    name-first      person.name.first-name
    name-last       person.name.last-name
    occupation      occupation.#TYPE
    ; preds
    pred attributive
    pred0 person
    pred1 occupation
  
```

```

predicate pred-birth
  variables
    req def person : c-person
    birth-event : c-birth-event
  properties
    birth-event == person.birth
  output
    template "{{name-first}} {{name-last}} was born in {{date-month}}/{{date-day}}/{{date-year}}".
    name-first      person.name.first-name
    name-last       person.name.last-name
    date-day        birth-event.date-instant.day
    date-month      birth-event.date-instant.month
    date-year       birth-event.date-instant.year
  
```

# OpenSchema: Biography Example

```
schema biography(self: c-person)
; name of the schema 'biography'
; self is the person the bio is about, required

; first paragraph, the person
plus
  pred-person(person|self)
optional
  pred-birth(person|self)
star ; zero or more aliases
  pred-alias(person|self)
star ; zero or more parents
  choice
    pred-father(self|self, parent|parent)
    pred-mother(self|self, parent|parent)
  star
    pred-person(person|parent)
star ; zero or more education
  pred-education(person|self)
paragraph-boundary
```

## Other Tools: Alusivo

- My most recent work has been in Regular Expression Generation
  - Particularly in evaluating REG algorithms over data that contains errors
    - Old versions of DBpedia
- As part of that work, I wrote a Java library that implements existing REG algorithms encoded as RDF inputs
  - Compatible with SimpleNLG
- <https://github.com/DrDub/Alusivo>

# Using Alusivo

```
List<URI> confusors = new ArrayList<URI>();  
URI confusor1 = f.createURI("http://alusivo/ballfar");  
URI confusor2 = f.createURI("http://alusivo/redballclose");  
confusors.add(confusor1);  
confusors.add(confusor2);  
URI referent = f.createURI("http://alusivo/redmiddle");
```

```
RepositoryConnection conn = rep.getConnection();  
URI balltype = f.createURI("http://alusivo/ball");  
URI color = f.createURI("http://alusivo/color");  
URI distance = f.createURI("http://alusivo/distance");  
conn.add(new StatementImpl(referent, RDF.TYPE, balltype));  
conn.add(new StatementImpl(confusor1, RDF.TYPE, balltype));  
conn.add(new StatementImpl(confusor2, RDF.TYPE, balltype));  
conn.add(new StatementImpl(referent, color, f.createLiteral("red")));  
conn.add(new StatementImpl(confusor1, color, f.createLiteral("black")));  
conn.add(new StatementImpl(confusor2, color, f.createLiteral("red")));  
conn.add(new StatementImpl(referent, distance, f.createLiteral("middle")));  
conn.add(new StatementImpl(confusor1, distance, f.createLiteral("far")));  
conn.add(new StatementImpl(confusor2, distance, f.createLiteral("close")));
```

# REG Algorithms in Alusivo

- Incremental Algorithm by Dale and Reiter
  - Uses a pre-determined property order
  - The “red close ball”
- Constraint Satisfaction Programming-based algorithm by Gardent
  - Able to generate negations
  - Example: distinguish *Paul McCartney* from *Ringo Starr, John Lennon, George Harrison*
    - NOT associated musical artist: *Plastic Ono Band*
- Graph algorithm by Krahmer, Van Erk and Verleg,
  - Based on graph isomorphisms.



# Alusivo: Research

- NAACL 2012:
  - DBpedia is useful enough for referring expressions.
- MICAI2015/WebNLG2016:
  - Using two versions of DBpedia allows studying impact of errors in referring expression generation.
- Iberamia 2016:
  - Robustness helps to learn Preference Ordering for properties for the Incremental Algorithm.
- SimBig 2017:
  - Predicting changes on DBpedia.
- Joint work with Dominguez, Barsotti, Estrella and others from UNC-Cordoba Argentina.

# Outline

- 1 Natural Language Generation
  - These Lectures
  - The NLG Problem
- 2 NLG Methods
  - NLG Architecture
  - NLG Subtasks
  - Linguistic Theories
- 3 Examples
  - Domains
  - Tools
  - Thoughtland

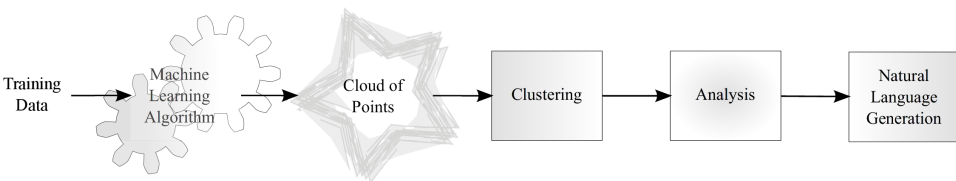
# Thoughtland

- From Duboue, 2013 “On the Feasibility of Automatically Describing  $n$ -dimensional Objects”

*This paper introduces the problem of generating descriptions of  $n$ -dimensional spatial data by decomposing it via model-based clustering. I apply the approach to the error function of supervised classification algorithms, a practical problem that uses Natural Language Generation for understanding the behaviour of a trained classifier.*

- <https://github.com/DrDub/Thoughtland>

# Thoughtland Architecture



# Thoughtland Input

- A small data set from the UCI ML repo, the Auto-Mpg Data:

<http://archive.ics.uci.edu/ml/>

[machine-learning-databases/auto-mpg/](http://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/)

```
@relation auto_mpg
@attribute mpg numeric
@attribute cylinders numeric
@attribute displacement numeric
@attribute horsepower numeric
@attribute weight numeric
@attribute acceleration numeric
@attribute modelyear numeric
@attribute origin numeric
@data
18.0,8,307.0,130.0,3504.,12.0,70,1
14.0,8,455.0,225.0,3086.,10.0,70,1
24.0,4,113.0,95.00,2372.,15.0,70,3
22.0,6,198.0,95.00,2833.,15.5,70,1
27.0,4,97.00,88.00,2130.,14.5,70,3
26.0,4,97.00,46.00,1835.,20.5,70,2
... +400 more rows
```

# Thoughtland Output

- MLP, 2 hidden layers (3, 2 units), acc. 65%:

*There are four components and eight dimensions. Components One, Two and Three are small. Components One, Two and Three are very dense. **Components Four, Three and One are all far from each other.** The rest are all at a good distance from each other.*

- MLP, 1 hidden layer (8 units), acc. 65.7%:

*There are four components and eight dimensions. Components One, Two and Three are small. Components One, Two and Three are very dense. **Components Four and Three are far from each other.** The rest are all at a good distance from each other.*

difference is **highlighted**

# Thoughtland Output

- MLP, 1 hidden layer (1 unit), acc. 58%, Thoughtland generates:

*There are five components and eight dimensions. Components One, Two and Three are small and Component Four is giant. Components One, Two and Three are very dense. Components One and Four are at a good distance from each other. Components Two and Three are also at a good distance from each other. Components Two and Five are also at a good distance from each other. The rest are all far from each other.*

# Thoughtland Component: Clustering

- The cloud of error points is clustered using a mixture of Dirichlet models
  - As implemented by Apache Mahout.
  - This clustering approach has a geometrical representation in the form of  $n$ -balls ( $n$ -dimensional spheres)
- Some input features present a natural geometric groupings which will opaque the error function.
  - The error coordinate is re-scaled using the radius of an  $n$ -ball that encompasses all the input features



# Thoughtland Component: Analysis

- For each  $n$ -ball: overall size, density, distances to the other  $n$ -balls.
  - Put the numbers into perspective.
- Example: density number of points in an  $n$ -ball given its volume:
  - $\text{density} > 10 \times \overline{\text{density}} \rightarrow$  very dense
  - $\text{density} > \overline{\text{density}} \rightarrow$  dense
  - $\text{density} < \frac{\overline{\text{density}}}{2} \rightarrow$  sparse

TYPE	COMPONENT(S)	VALUE
Size	0	Big
Distance	0,1	Big
Distance	0,2	Big
Distance	0,3	Medium
Distance	0,4	Big
Distance	0,5	Big
Size	1	Big
Distance	1,2	Big
Distance	1,3	Big
Distance	1,4	Big
Distance	1,5	Big
Size	2	Very Big
Density	2	Small
Distance	2,3	Big
Distance	2,4	Big
Distance	2,5	Big
Size	3	Small
Density	3	Very Big
Distance	3,4	Big
Distance	3,5	Big
Size	4	Small
Density	4	Very Big
Distance	4,5	Big
Size	5	Small

# Thoughtland Component: Planner


- Two schemata:
  - Components are presented in order .
  - Attributes are presented in order (pictured next).
- The system presents the user the shorter description.

# Thoughtland Component: Schemata

```
schema by-attribute(whole: c-full-cloud)
; first sentence, overall numbers
pred-intro(cloud|whole)
aggregation-boundary
star
  pred-size()
aggregation-boundary
star
  pred-density()
aggregation-boundary
star
  pred-distance()

predicate pred-density
variables
  req def component : c-n-ball
  req attribute : c-density
properties
  component == attribute.component
output
  pred has-attribute
  pred0 component
  pred1 attribute
  pred2 magnitude
```

# Thoughtland Component: Interface



**Thoughtland**

*I spoke not of a physical Dimension,  
but of a Thoughtland whence, in theory,  
a Figure could look down upon Flatland  
and see simultaneously the insides of  
all things*

**Submit a Weka ARFF file for analysis**

Algorithm to use:

Tue Mar 12 20:59:54 EDT 2013 There are five components and eight dimensions. Component four, component two and component three are small and component one is giant. Component three, component four and component two are very dense. The components one and two are far from each other. The components one and four are far from each other. The components one and five are far from each

# Summary

- NLG at its core deals with **representing and handling large number of principled decisions**.
  - A process of enriching the input representation culminating into a full fledged text.
- Not necessarily the inverse of NLU.
  - Deals with communicative intention much more than NLU.
- Can be done with success for limited domains.
  - Tons of work, very hand crafted.
- Next lecture:
  - Using data.
  - Evaluation.

## For Further Reading



E. Reiter, R Dale .

*Building Natural Language Generation Systems.*

MIT Press, 2000.



A. Gatt, E. Krahmer.

*Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation.*

Journal of Artificial Intelligence Research 61 65-170, 2018.