Survival skills for seminar attendants

Simon Razniewski

Date

https://doodle.com/poll/q8z8grgz4qbqn3ie

- **December 13, 2017, 23:59** -- students send a suggestion of the outline of their seminar paper, including an itemization of the planned content for each section.
- **January 31, 2018, 23:59** -- students submit their final seminar paper.
- **February 20, 2018, 23:59** -- students send preliminary slides
- **March 3, 2018, 23:59** -- students send their final slides which they will use in the block seminar
- **March 6, 2018** -- Block seminar, Day 1
- **March 7, 2018** -- Block seminar, Day 2

Outline

- KB cheatsheet
- How to…
    1. Research a topic
    2. Read a paper
    3. Write a report
    4. Give a seminar talk
- Topic assignment

Outline

- **KB cheatsheet**
- How to…
    1. Research a topic
    2. Read a paper
    3. Write a report
    4. Give a seminar talk
- Topic assignment

KB cheatsheet - terminology

- **Fact**/statement/claim/triple: Atomic building block
  locatedIn(Paris, France)
- **Entities**: Objects about which statements can be made
  Paris; Trump; Irony
- **Property**/predicate/relation/attribute: Express can be said
  locatedIn, worksAt, antonymOf
  - Literals: Attribute values that are no entities
    1.63m; 54.85° N
- **Qualifiers** for facts: Time, location, references, …
- **Types**/classes: Allow to group similar entities
  President, noun, Greek god
- **Type/property hierarchy**: Tree-like hierarchy among types/properties (cf. inheritance in object-oriented programming)

KB cheatsheet – aspects (1/2)

1. What is the topic?
   - Encyclopedic
   - Procedural
   - Common-sense
2. What is the KB used for?
   - Master-data
   - Question answering
   - …
3. How is it created?
   - Manual
     - Experts
     - Crowd
   - Automated
     - Knowledge harvesting
     - Text extraction
   - Machine learning on top of existing facts

KB cheatsheet – aspects (2/2)

4. Content
   - What are the entities
     - Abstract concepts (tree, rain, envy)
     - Concrete concepts (Trump, Dudweiler, Beatles)
   - Is there a class/predicate hierarchy?
   - What are the relations?
     - Few curated ones (YAGO)
     - Open-ended (open text extraction)
5. How is data modelled?
   - Object identity
     - Unique object identifiers (YAGO)
     - Weak object identity (text extraction)
   - Relation format
     - Binary/n-ary relations
   - Are there constraints?
     - Soft constraints
     - Hard constraints
   - Can the KB cope with the dynamicity of knowledge?
     - Does the modelling allow to represent a temporal dimension?
     - Is there an update mechanism for the KB?
6. How is data accessible
   - Data exports (RDF, JSON, …)
   - Web interfaces?

Outline

- KB cheatsheet
- **How to…**
  1. **Research a topic**
  2. Read a paper
  3. Write a report
  4. Give a seminar talk
- Topic assignment

Topic research: Foundations

- **Goal**: Find relevant literature
- **Input**: One or two reference papers
- **Idea**: Greedy traversal of a relatedness graph ("snowballing")

- What gives related papers?
  - Papers that are cited
  - Papers that cite the work
    - Sources: Google Scholar, digital libraries (ACM digital library, Springer, Elsevier)
  - Other work of same authors
    - Sources: Institute/personal webpages, DBLP
  - Tool or project websites (e.g., AMIE)
  - Conference/journal/workshop webpages
  - Keyword-based search (Google Scholar)

Greedy search heuristics

- How to judge importance of a paper?
  - #citations
  - Venue
    - A*, A, (B), …
    - http://portal.core.edu.au/conf-ranks/
  - Paper type and length
    - Full/short/poster/Arxiv
  - Authors

Keeping track

- Method
  - Mendeley
  - Google Scholar “My library”
  - Text file

- Take note of narrative patterns
  - “The classic rule mining”, “The Semantic Web community”, “The Paris people”
  - Humans love narratives
  - Helps to remember

Demo

1. Google “AMIE: association rule mining under incomplete evidence in ontological knowledge bases”
2. Google Scholar
3. See in and outgoing citations
4. See author’s DBLP
5. CORE

Outline

- KB cheatsheet
- **How to…**
    1. Research a topic
    2. **Read a paper**
    3. Write a report
    4. Give a seminar talk
- Topic assignment

Research paper: Common structure

- Abstract
- Introduction
- Related Work
- Background/Formalization
- Methodology (often specific name)
- Experimental Setup
- Evaluation
- Discussion
- Conclusion

What to find where

- Abstract contains everything ☺
  - Introduction contains more of everything ☺
    - Both mention what is novel
- Related work great to discover what to read next
- Background/formalization may contain surprising assumptions/simplifications
  - *"for the remainder of the paper, we assume that all text sources only contain true statements"*
- Experimental setup says how method was evaluated
- Discussion/conclusion contain limitations and open questions

How to read a paper

- Read abstract
- Read introduction
- Skim rest, especially methodology and results
- Read introduction again
- Decide further process
  - Continue reading
  - Read previous work
  - Discard
- If you don’t understand something
  - Read a similar paper: Might explain better
  - Take pen&paper and simulate a scenario
  - Look for theses/journal articles

Outline

- KB cheatsheet
- **How to…**
    1. Research a topic
    2. Read a paper
    3. **Write a report**
    4. Give a seminar talk
- Topic assignment

General writing

- Follow the Latex template (course website)
  - Miktex and other Latex distributions
  - Lyx (Word-style editing)
  - Overleaf/Sharelatex (Google-docs-like online editing)
- 11-13 pages content
- Do not plagiarize

Structure

- Use a standard outline
  - Introduction
  - Wider area/history of the topic
  - Foundations/assumptions/specific scenario the work looks at
  - Technical parts
    - Method 1, Method 2, Other methods
  - Critical evaluation and discussion
  - Conclusion

→ Helps the standard reader in retrieval

Watch your writing

- Does only the content matter?

- Surface features like appearance highly influence other dimensions of evaluation
- Known as the **Halo effect** since almost 100 years (Thorndike, 1920)
- Academic publishing:
  3 typos in the abstract = rejection

Assignment 1: Hamming Code

Name: [redacted]
Student ID: [redacted]

The code in *HammingCode.java* contains the two major functions *encode()* and *decode()* as well as some auxiliary methods that will be explained further in this paper.

First of all, the method *encode(String message, String filename)* takes as parameters a string containing a message that will be encoded using a (7,4)-Hamming Code and another string specifying the file path where the result will be stored.
The algorithm proceeds in a way that one character after the other is taken from the message string, gets processed and is stored in bit representation in the specified file before it is the next character's turn.
At the beginning, a character is converted to a binary string (a String object containing only 0s and 1s). Since most characters may be converted to 7-bit strings and we need 4-bit blocks to apply Hamming Code in a later moment, we add one or more 0s at the beginning of the string in order to make it 8 bits long. Then the result string is split in half and method *applyHammingCode(String input)* is called on both halves. In this method, the actual computations for Hamming Code take place and the parity bits are calculated. Since the single bits of the 4-bit input string are converted to integers, the parity bits can simply be computed by adding up the necessary data bits and using "modulo 2" on the sum. Finally, the method returns an array of integers, containing all seven bits (4 data bits + 3 parity bits) of a Hamming block in the right order (p_1, p_2, d_1, p_3, d_2, d_3, d_4). Then, the values of this array are written to the file and the next character of the input message is processed.
The converting of characters to integers mentioned before, is done in another auxiliary method called *charToInt(char character)* that simply takes a character variable as parameter and checks whether it contains "1". If so, 1 as numeric value is returned or 0 otherwise.

The method *decode(String filename)* reads a 7-bit block from the specified file, then filters out the 4 data bits and re-calculates the parity bits for these 4 bits. If the newly calculated parity bits are equal to the parity bits read from the file, it means that all data bits were transmitted correctly and no error correction has to be done. Otherwise, if the parity bits read from the file and the ones calculated match either in only 1 bit or if they do not match at all, there must have occurred a single bit error, which can be corrected easily. Using the rules to calculate the parity bits, the method first finds out which data bit was transmitted incorrectly, then calls method *flipBit(String input, int index)*, which simply flips the bit at the specified index. This means, if a bit had value 1 in the first place, it is assigned 0 now and vice versa.
Now that an eventual error has been corrected, the program checks whether the current bit block belongs to the first or to the second half of a character's bit representation. If it is part of the first half, it is stored in a variable for later computations and a new 7-bit block is read from the file and processed. Otherwise, the data bits of the first and second half are connected and converted to a character. Then this character is appended to a string and the next 7-bit block representing a character's first half is read from the file. When all bits of the file have been processed, the string containing the characters is returned and printed to the console.

[redacted]

RMI-Chat Report
Assignment 3

1. Description of the algorithm

First, I implemented two interfaces:

- ServerInterface, which contains the methods addClient(ClientInteface client, String username), sendMessage(String message) and disconnect(ClientInterface client);
- ClientInterface, which contains the method getMessage(String message).

Then, I developed for the server point of view a class: the Server class.

The Server class is the class used to create the Server and it implements the ServerInterface. In the Server class, the main method sets the security manager and calls the Server(JFrame currentFrame) constructor. This method opens a JFrame displaying the IP address of the server (that will be then inserted by a non-local client to connect), creates a remote object using the port "1099" and a server named "RMIChatServer". It also calls another Server constructor, that instantiates two ArrayLists, "clients" of ClientInterfaces and "usernames" of Strings, which will then contain respectively the connected clients and their usernames.

The Server class contains also the ServerInterface's methods overridden:

- addClient(ClientInterface client, String username), which adds the two parameters to the right ArrayLists;
- sendMessage(String messsage), which sends the parameter "message" to all the clients stored in the "users" ArrayList through the getMessage(String message) method;
- disconnect(ClientInterface client), that identifies the username of the disconnecting client, removes the user to both ArrayLists, creates the "user disconnected" message and sends it through the sendMessage method.

Eventually, I developed for the client point of view three classes: the Client class, the WelcomeWindow class and the ChatWindow class.

The Client class connects the client to the server and manages the graphical Windows. Its main method contains only the call to the WelcomeWindow's constructor, which starts the user's graphical interface (and that will then return the flow to this Client class).

There are two constructors. One of them is Client(String address), which sets the security manager and connects to the RMI Server; then opens the ChatWindow and calls the other Client constructor.

Good presentation

- Structure!
- No typos!
- Pictures!
- Examples!
- Correct and concise language!

How to write a good paper

- Iterate…

Outline

- KB cheatsheet
- **How to…**
    1. Research a topic
    2. Read a paper
    3. Write a report
    4. **Give a seminar talk**
- Topic assignment

How to give a seminar talk

- General:
  - Presentations are the 15-minute fame of researchers
  - Utmost important
  - Good presenting is no God-given skill but a hard-learned craft

- Important points
  A. Talks live or die with the example(s) used
  B. Do not try to say everything
  C. Less content per slide
  D. Avoid being a tool-fool

A. Examples

- Think about them carefully
  - Funny and/or interesting case
  - Bonus: References to local/current situation/joint background
    - Talk in Paris: Eiffel tower, talk in Germany now: failed coalition talks, …
- Need to be consistently used
  - Not: on Slide 3 “Mary lives in Munich”, on Slide 10 “Mary lives in Berlin”
  - Ideal: Same examples for the whole presentation (extended piece by piece)
- If possible: 1 use of blackboard per presentation

B. Do not say everything you know

- Most situations: Talk is a teaser
  - “This is an interesting problem”
  - “I have a solution for …”
  - “I can significantly improve over the state of the art”
  - …
- Main goal is not to convey the technical content
  - That’s what papers/reports are for
- Often: Explain one favorite module of the approach in technical detail

B. Adjust to the audience

- Adjust talk to audience knowledge

- Audience rarely have same technical background

→ Don’t outdistance them

- But don’t bore them either
  - E.g. in the context of this seminar, don’t repeatly elaborate “a knowledge base is a set of triples (s,p,o), …”

- Adjust to context
  - We have just seen …, now we …
  - Look what other talks might cover – talk to these presenters!

C. Less content per slide (1/2)

- Most slides contain too much content
  - 1-2 minutes per slide
- Whenever looks like too much
  - Split in two
  - Use animation effects to remove content not needed anymore (e.g. in examples)
- Font size not below 20 (PPT)
  - This is 28, quite big
  - This is 24, reasonably OK
  - This is 20, which should be your limit
  - This is 16, what do you think people in the last row see?

C. Less content per slide (2/2)

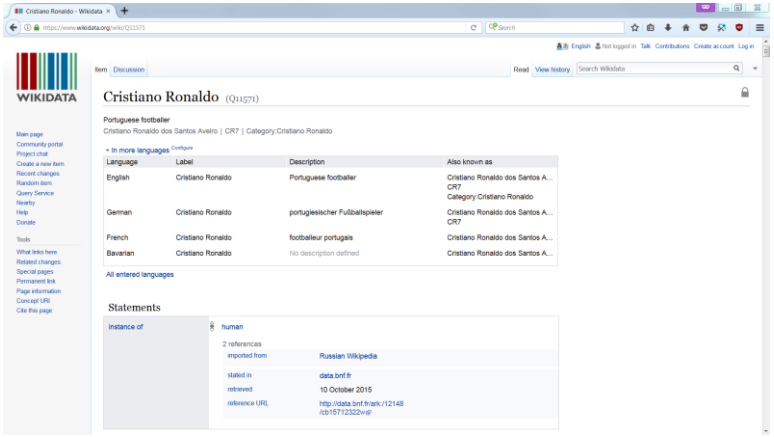
- But this is not only about font size
- Even though I use font size 28/20 for all text on this slide, you may get the impression that something is wrong with this slide
  - Is this possibly related to the amount of information conveyed?

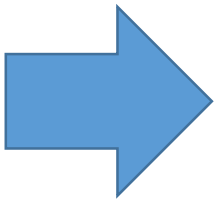
  →Well, actually, what I am saying is not very deep. You already know very well that the human brain has only a limited attention span. So by the time you are reading this you certainly have forgotten what was written at the top of this slide.
- More structure does not help either
  - Because too much is simply too much. So really pay attention both to font size and to amount of information conveyed

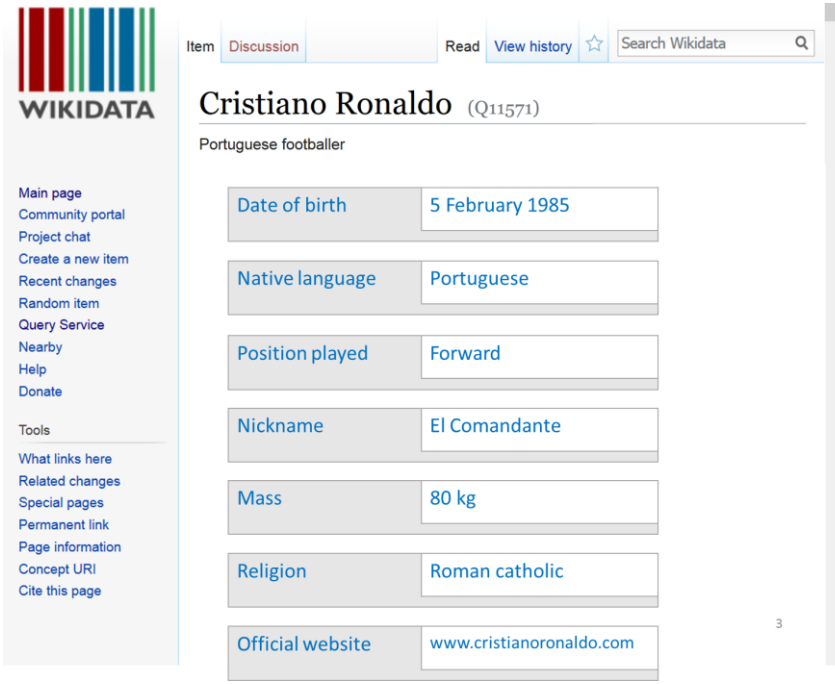
D. Avoid being a tool-fool

- Latex vs. Powerpoint
- Redraw diagrams
- Modify images

Modify figures where needed







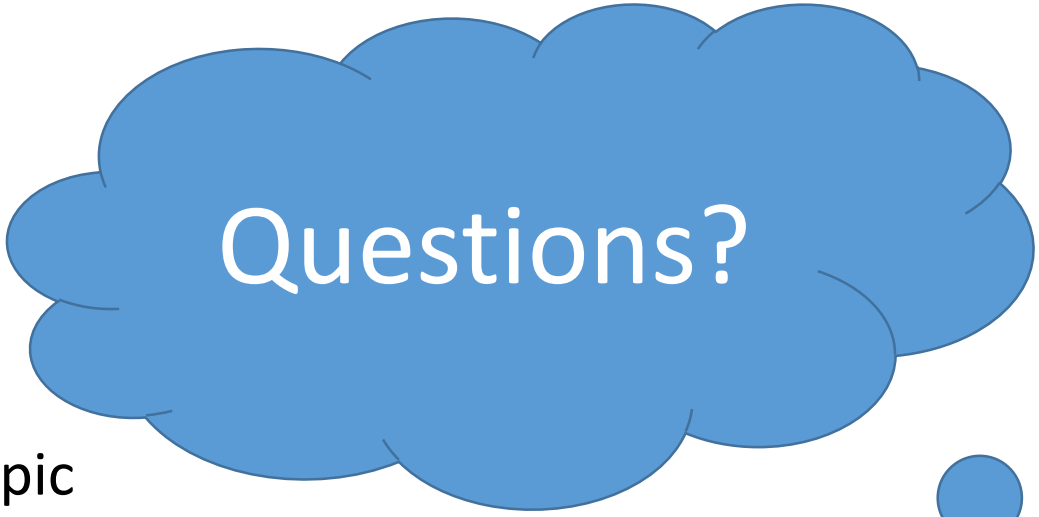
Varia (1/3)

- Page numbers
- Avoid useless info (name, title, date) on slides
- No empty last slide “questions”,
  - Rather overlay over conclusion

  → Gives audience an anchor

Questions?

Summary



- KB cheatsheet
- How to…
  1. Research a topic
     - Use multiple means
  2. Read a paper
     - Read, read, read
  3. Write a report
     - Watch the simple things
  4. Give a seminar talk
     - Less is more
     - Spend a lot of effort on good examples
- Topic assignment

Varia (2/3)

- Tell three times
  - Tell what you are going to tell
  - Tell it
  - Tell what you told
- Easy way: Repeat outline throughout talk
  - Good point to breathe in deeply, take a sip of water, look at your watch, ask “everything clear so far?”

Varia (3/3)

- One slide: “Not in this talk”

Not in this talk

- How to typeset math formulas
  - See e.g. “Mathematical writing” by Donald Knuth
- How to write good language
  - Search online
- Rhetorical hints
  - Take a course at the Center for Key Competencies and University Didactics

Practice (1/3)

- John Wooden's 8Ps of Success
    1. Plan
    2. Prepare
    3. Practice
    4. Practice
    5. Practice
    6. Practice
    7. Practice, and
    8. Practice

Practice (2/3)

- Everybody is nervous
- Enjoy your presentation, and your audience will enjoy
- Keep visual contact
- Take questions
  - Acknowledge gaps, take issues offline
  - Help answering questions

Practice (3/3)

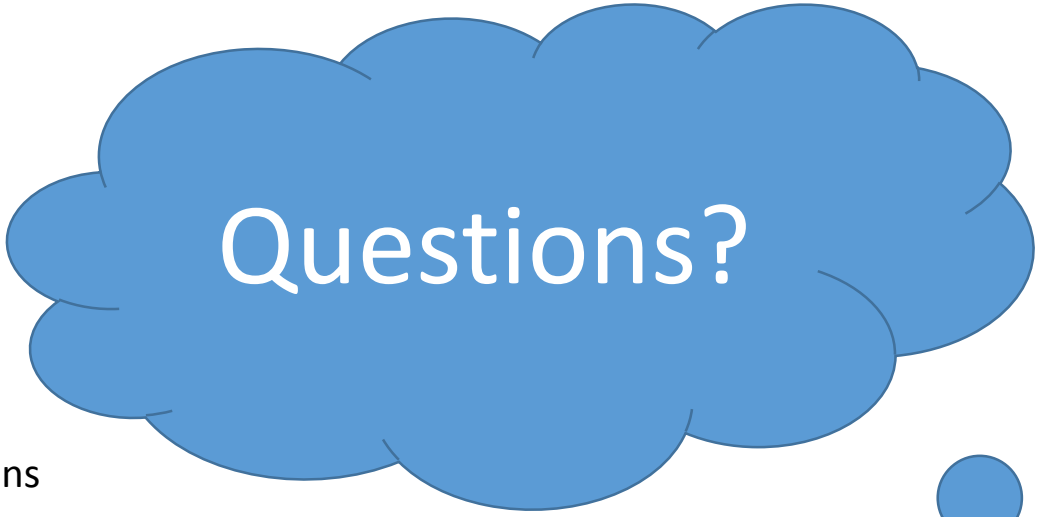
- Same setting is not required
  - Explain over lunch to a colleague
  - Your parents are OK too
- Helps to get more familiar
- Watch for flow, consistency, odd steps

Outline

- KB cheatsheet
- How to…
    1. Research a topic
    2. Read a paper
    3. Write a report
    4. Give a seminar talk
- **Topic assignment**

1: Collaborative KBs: Wikidata	Safura Isayeva
2: Structured information extraction: DBpedia and YAGO	Mang Zhao
3: KB population in the TAC 2016 challenge	Nitisha Jain
4: General common-sense KBs: ConceptNet and WebChild	Xueting Li
5: Domain-specific activity KBs	Anu Goel
6: The Allen AI science challenge & building a science KB: Aristo	Frederik Schmitt
7: KB association rule mining	Joscha Cüppers
8: Inferring new facts with vector space embeddings	Harshita Jhavar
9: Exploring and profiling KBs	Adrian Spirescu
10: KB question answering	Shrestha Ghosh
11: Hybrid question answering using KBs and text	Aydan Rende
12: Non-encyclopedic QA in the science domain	Khansa Rekik
13: Coreference resolution	Damyana Gateva
14: Referent prediction	Natalie Wirth

Summary



- KB cheatsheet
- How to…
  1. Research a topic
     - Use multiple means
  2. Read a paper
     - Read, read, read
  3. Write a report
     - Watch the simple things
  4. Give a seminar talk
     - Less is more
     - Spend a lot of effort on good examples
- Topic assignment

- Next steps
  - Enroll in HISPOS
  - Prepare your outline till 13.12.
  - Meet us