— INF4820 —
Algorithms for Artificial Intelligence and Natural Language Processing

Introduction and Overview

Erik Velldal & Stephan Oepen

Language Technology Group (LTG)

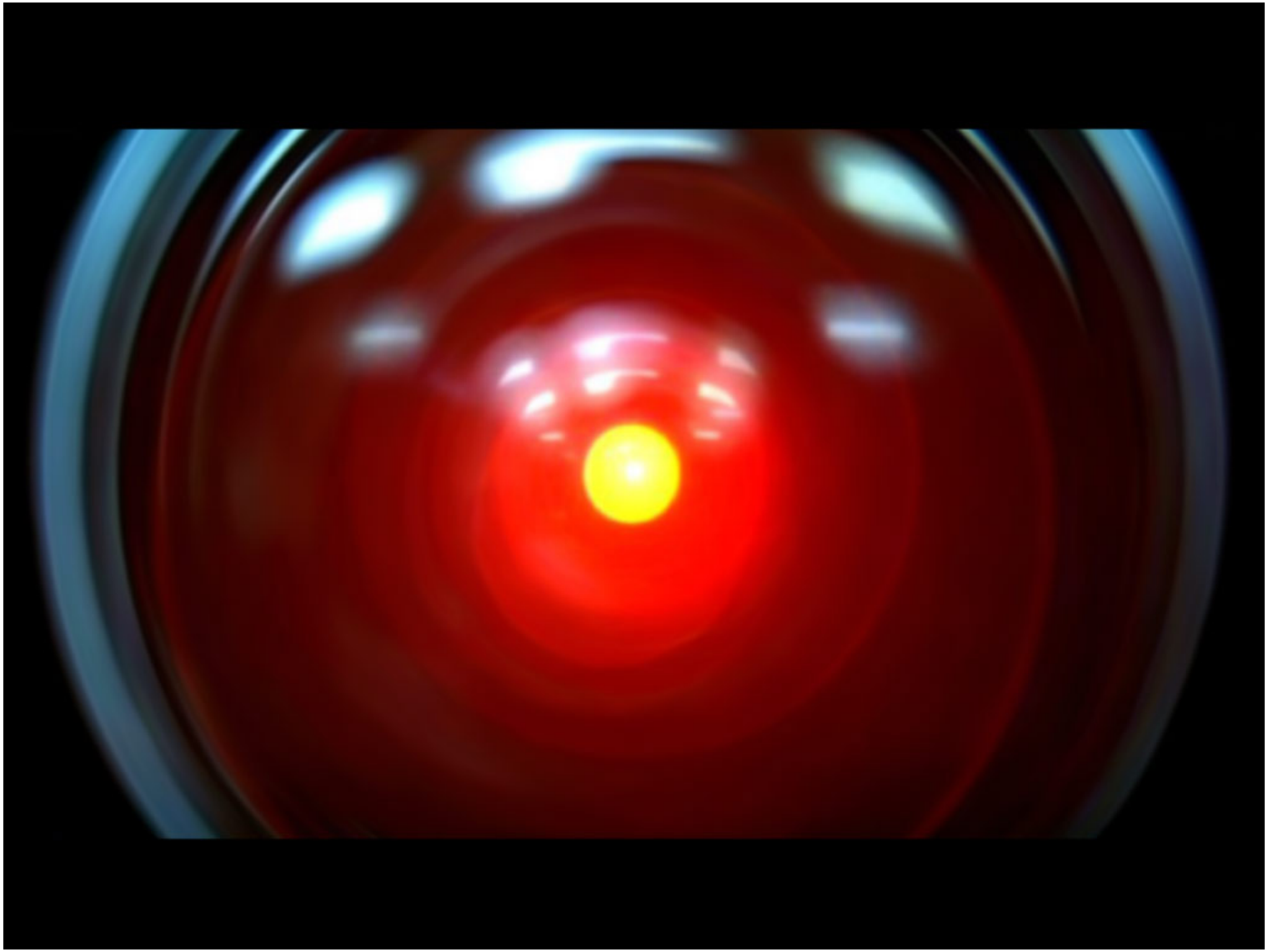
August 19, 2015

Topics for today

Overview

- Course motivation and introduction:
- AI, NLP, ML — What are they?
- Lisp — What and why?
- Outline of lectures and learning goals.
- Practical details.

What is AI?



HAL 9000 in “2001: A Space Odyssey” (1968)

AI, hype, and the end of the world

- AI has always been subject to cycles of hype.

AI, hype, and the end of the world

- AI has always been subject to cycles of hype.
- A lot of media attention recently concerning the existential risk of AI:

AI, hype, and the end of the world

- AI has always been subject to cycles of hype.
- A lot of media attention recently concerning the existential risk of AI:
- Stephen Hawking et al. in an op-ed in the Independent (spring 2014):
  - *it's tempting to dismiss the notion of highly intelligent machines as mere science fiction. But this would be a mistake, and potentially our worst mistake in history.*
  - *Whereas the short-term impact of AI depends on who controls it, the long-term impact depends on whether it can be controlled at all.*

AI, hype, and the end of the world

- AI has always been subject to cycles of hype.
- A lot of media attention recently concerning the existential risk of AI:
- Stephen Hawking et al. in an op-ed in the Independent (spring 2014):
  - *it's tempting to dismiss the notion of highly intelligent machines as mere science fiction. But this would be a mistake, and potentially our worst mistake in history.*
  - *Whereas the short-term impact of AI depends on who controls it, the long-term impact depends on whether it can be controlled at all.*
- Elon Musk (Tesla Motors, SpaceX) at a MIT talk (fall 2014):
  - *With AI we are summoning the demon.*

The early years

- Alan Turing, 1950:
  - *I propose to consider the question, 'Can machines think?'*

The early years

- Alan Turing, 1950:
  - *I propose to consider the question, 'Can machines think?'*
- The term 'AI' coined by John McCarthy (Dartmouth Conference, 1956).
  - *The science and engineering of making intelligent machines.*
  - *Every aspect of learning or any other feature of intelligence can be so precisely described that a machine can be made to simulate it.*

The early years

- Alan Turing, 1950:
  - *I propose to consider the question, 'Can machines think?'*
- The term 'AI' coined by John McCarthy (Dartmouth Conference, 1956).
  - *The science and engineering of making intelligent machines.*
  - *Every aspect of learning or any other feature of intelligence can be so precisely described that a machine can be made to simulate it.*
- Language always had a central place, cf. the Turing Test.

Bag of tricks

- The early years: simple chatbots, expert systems, game playing (chess), theorem proving, blocks world, ...

Bag of tricks

- The early years: simple chatbots, expert systems, game playing (chess), theorem proving, blocks world, ...
- Moving target: Whatever requires 'intelligent' decisions, but seems out of reach technologically?

Bag of tricks

- The early years: simple chatbots, expert systems, game playing (chess), theorem proving, blocks world, ...
- Moving target: Whatever requires 'intelligent' decisions, but seems out of reach technologically?
  - Web search arguably would have been AI a couple of decades ago.
  - Open-domain machine translation out of reach until around 2005.

Bag of tricks

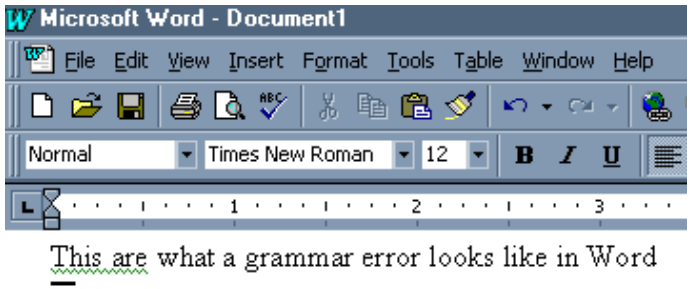
- The early years: simple chatbots, expert systems, game playing (chess), theorem proving, blocks world, ...
- Moving target: Whatever requires 'intelligent' decisions, but seems out of reach technologically?
  - Web search arguably would have been AI a couple of decades ago.
  - Open-domain machine translation out of reach until around 2005.
- For our purpose: AI is a toolkit of methods for problem solving and representation.

What is Natural Language Processing?

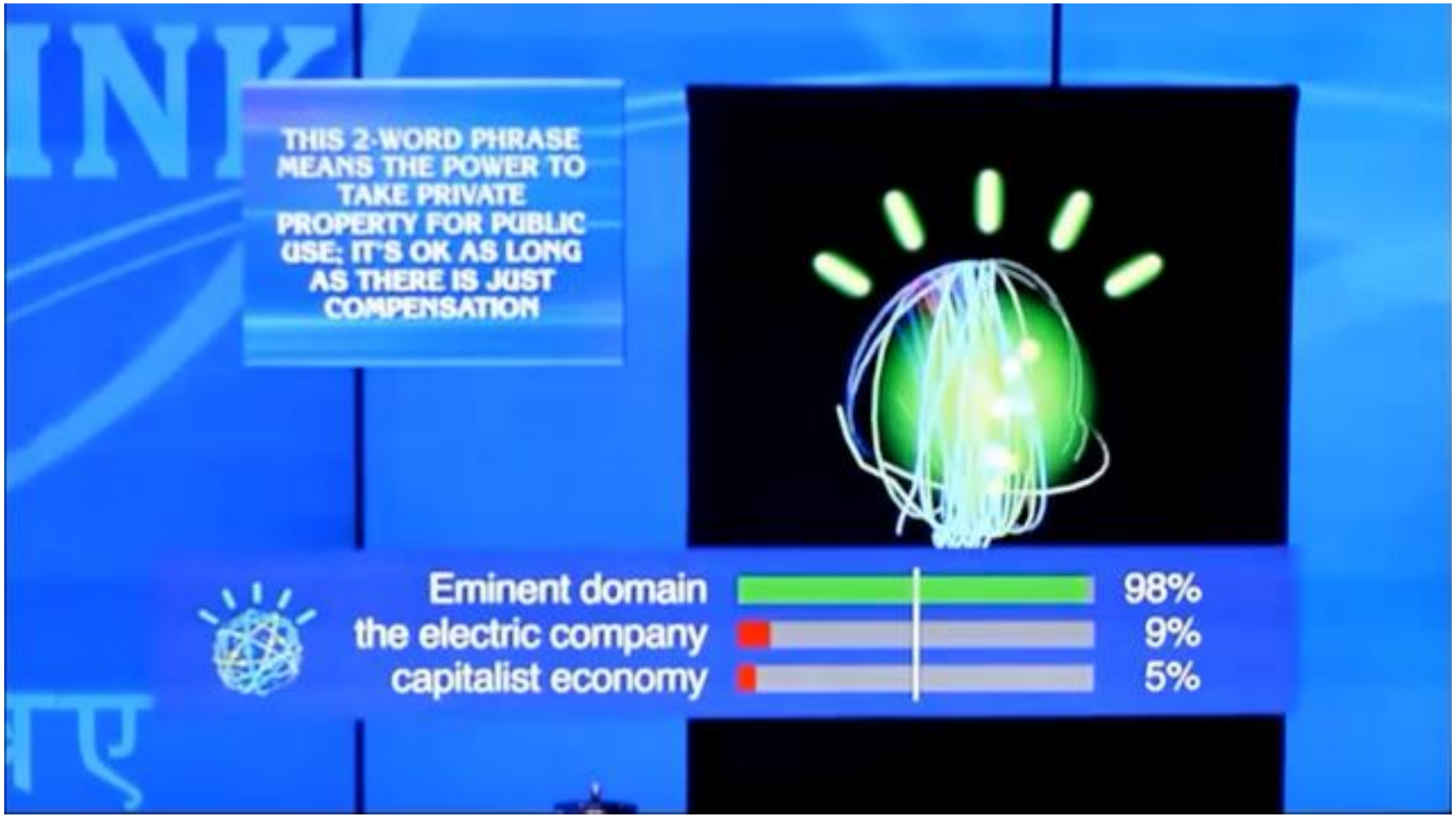
- Making computers 'understand' human language
- Aka language technology or computational linguistics
- Young and interdisciplinary field:
- Computer Science + Linguistics
- (+ Cognitive Science + Statistics + Information Theory + Machine Learning + . . .)

Some applications

- Grammar and/or spell checkers, auto-completion
- Machine translation
- Q&A systems
- Dialog systems
- Speech recognition and synthesis
- Intelligent information extraction
- Summarization
- Sentiment analysis
- Any application requiring an understanding of language...



A recent landmark



IBM Watson beats long-time *Jeopardy!* champions, 2011

What makes NLP hard?

Ambiguity

- I.e. the property of being open to multiple interpretations.
- All levels of linguistic description are associated with ambiguities.

What makes NLP hard?

Ambiguity

- I.e. the property of being open to multiple interpretations.
- All levels of linguistic description are associated with ambiguities.
- For humans, ambiguity is a feature: language is an efficient code.
  - The same expressions can be re-used in different contexts.
  - Most of the information can be underspecified.
  - Interpretation relies on background knowledge and our expectations in a given context of use.

What makes NLP hard?

Ambiguity

- I.e. the property of being open to multiple interpretations.
- All levels of linguistic description are associated with ambiguities.
- For humans, ambiguity is a feature: language is an efficient code.
  - The same expressions can be re-used in different contexts.
  - Most of the information can be underspecified.
  - Interpretation relies on background knowledge and our expectations in a given context of use.
- Disambiguation is a central problem in NLP

What makes NLP hard?

Ambiguity

- I.e. the property of being open to multiple interpretations.
- All levels of linguistic description are associated with ambiguities.
- For humans, ambiguity is a feature: language is an efficient code.
  - The same expressions can be re-used in different contexts.
  - Most of the information can be underspecified.
  - Interpretation relies on background knowledge and our expectations in a given context of use.
- Disambiguation is a central problem in NLP $\longrightarrow$ Search problems.

Ambiguity: Some examples

Word level ambiguity

- Norwegian: *rett*.
- English:

Ambiguity: Some examples

Word level ambiguity

- Norwegian: *rett*.
- English: *meal*, *dish*, *straight*, *correct*, *fair*, *justice*, *right*, *court*, *law*, *direct*, *grade*, . . . ?

Ambiguity: Some examples

Word level ambiguity

- Norwegian: *rett*.
- English: *meal*, *dish*, *straight*, *correct*, *fair*, *justice*, *right*, *court*, *law*, *direct*, *grade*, . . . ?
- Ambiguous in meaning + syntactic category (part of speech).
- Need context to decide.

Ambiguity: Some examples

Word level ambiguity

- Norwegian: *rett*.
- English: *meal*, *dish*, *straight*, *correct*, *fair*, *justice*, *right*, *court*, *law*, *direct*, *grade*, . . . ?
- Ambiguous in meaning + syntactic category (part of speech).
- Need context to decide.

De hadde laget en deilig rett av grønnsaker.

Streken må være rett.

Kunden har alltid rett.

Du har rett til en advoktat.

Det er lovlig i henhold til norsk rett.

Slikt skjer rett som det er.

Vennligst rett disse prøvene!

Vi kjørte rett hjem.

Ambiguity: Some examples

Referential ambiguity

The authorities jailed the protesters because *they* { *advocated revolution.* / *feared revolution.*

Ambiguity: Some examples

Referential ambiguity

The authorities jailed the protesters because they { *advocated revolution.* / *feared revolution.*

Sentence-level ambiguity

I like eating sushi with { *tuna.* / *sticks.*

Ambiguity: Some examples

Referential ambiguity

The authorities jailed the protesters because they { *advocated revolution.* / *feared revolution.*

Sentence-level ambiguity

I like eating sushi with { *tuna.* / *sticks.*

Acoustic ambiguity

Let's talk about how to { *recognize speech* / *wreck a nice beach*

Some history: Different approaches to AI and NLP

- Traditionally, two broad paradigms in NLP (and AI).
  - The rationalist approach, based on hand-crafted formal rules and manually encoded knowledge.
  - The empiricist approach, based on automatically inferring statistical patterns from data.

Some history: Different approaches to AI and NLP

- Traditionally, two broad paradigms in NLP (and AI).
  - The rationalist approach, based on hand-crafted formal rules and manually encoded knowledge.
  - The empiricist approach, based on automatically inferring statistical patterns from data.
- 1950s – 80s: Rationalist / rule-based approaches.
- Late 1980's: Empirical systems outperform rule-based in the area of speech recognition.
- 1990s: NLP as whole sees a shift of interest from rationalist towards empirical approaches.
- 2000s: No longer conceived as opposing poles, but complementary approaches typically used together.

The basis of empirical methods

- The theoretical foundations are studied within the field of machine learning (ML) or statistical learning theory.

Machine Learning

... *the study of computer algorithms that improve automatically through experience* (Tom Mitchell 1997).

- Goal: Learn from examples, to make predictions about new data.

The basis of empirical methods

- The theoretical foundations are studied within the field of machine learning (ML) or statistical learning theory.

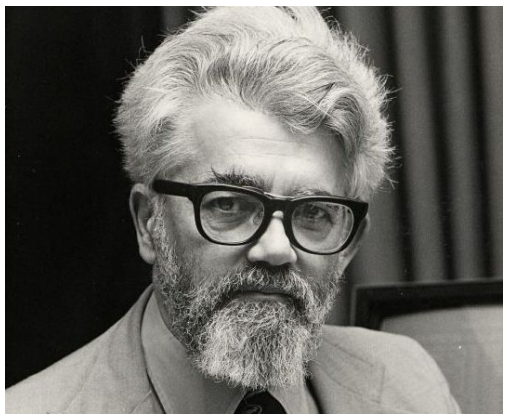
Machine Learning

... *the study of computer algorithms that improve automatically through experience* (Tom Mitchell 1997).

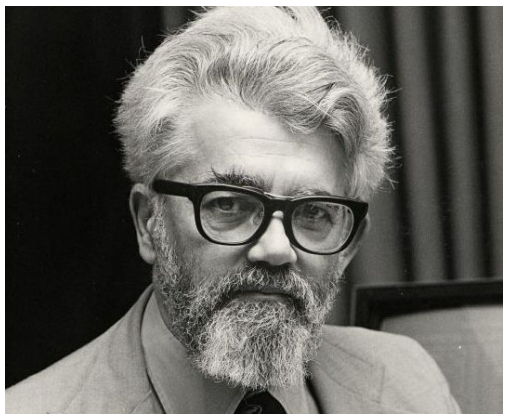
- Goal: Learn from examples, to make predictions about new data.
- Has applications in many other data-intensive sciences besides NLP, e.g. meteorology, biology, physics, robotics, signal processing, etc.
- An arsenal of methods: decision trees, support vector machines, maximum entropy models, naïve Bayes classifiers, artificial neural networks, genetic algorithms, ...

Lisp

- Powerful high-level language with long traditions.
- Especially strong support for symbolic and functional programming.
- "Discovered" by John McCarthy in 1958.

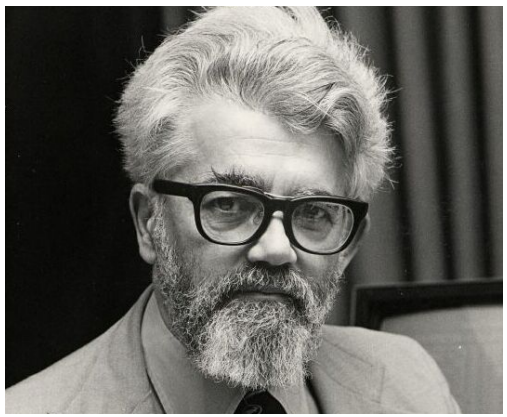


Lisp



- Powerful high-level language with long traditions.
- Especially strong support for symbolic and functional programming.
- "Discovered" by John McCarthy in 1958.
  - Initially intended as a mathematical formalism.
  - Then one of his students, Steve Russell, implemented an interpreter for the formalism, and Lisp the programming language was born.

Lisp



- Powerful high-level language with long traditions.
- Especially strong support for symbolic and functional programming.
- "Discovered" by John McCarthy in 1958.
  - Initially intended as a mathematical formalism.
  - Then one of his students, Steve Russell, implemented an interpreter for the formalism, and Lisp the programming language was born.
- Rather than Lisp becoming outdated, the tendency has been that other languages have developed towards Lisp.

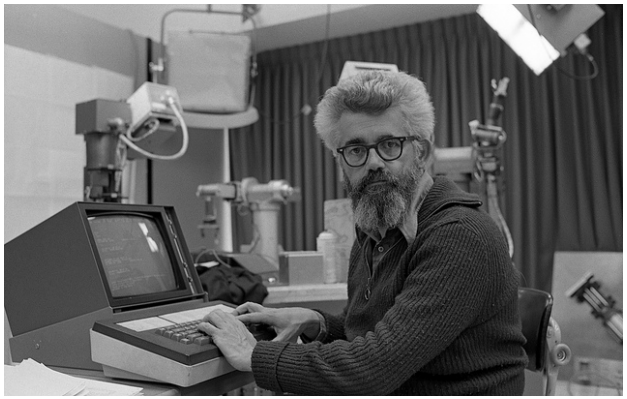
Common Lisp

```
(print "Hello world!")
```

- Several dialects, we will be using Common Lisp.
- Fully ANSI-standardized and stable.
- Rich language: multitude of built-in data types and operations.
- Easy to learn:
  - extremely simple syntax,
  - straightforward semantics.

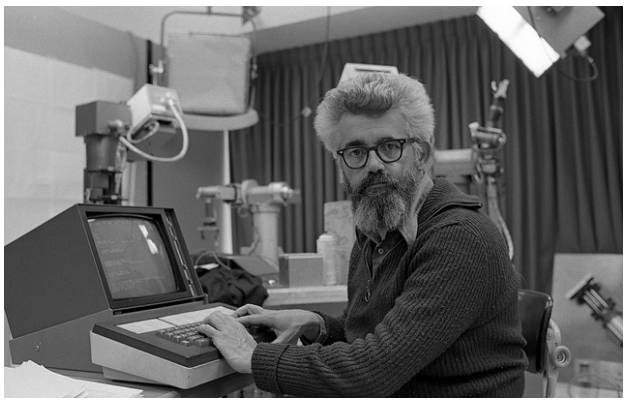
A note on Lisp and AI

- Often hailed (or dismissed) as "the AI language".
- While not quite true, there are several reasons for this coupling:
- AI coined by McCarthy in the mid-1950s.
- Lisp conceived by McCarthy in the mid-1950s.



A note on Lisp and AI

- Often hailed (or dismissed) as "the AI language".
- While not quite true, there are several reasons for this coupling:
- AI coined by McCarthy in the mid-1950s.
- Lisp conceived by McCarthy in the mid-1950s.
- In addition to being fast and powerful, Lisp is particularly well suited for:
  - Explorative programming
  - Rapid prototyping
  - Incremental and interactive development
  - Extending the language itself



Lisp + Emacs = a good match

- Steep learning curve, but with a big payoff:
- Emacs is an unusually powerful editor.
- Written in Emacs Lisp.
- Highly customizable—the Emacs Lisp dialect is also used as an extension language.
- Different “modes” make Emacs sensitive to different editing needs, e.g. depending on the specific programming language used.
- Prerequisite for an enjoyable Emacs experience: Spend some time mastering basic key commands!

Obligatory exercises

- Three obligatory exercises:
- Exercise 2 and 3 has two parts each:
- Five problem sets in total.
- In order to pass and qualify for the exam you need a least
  - 6 of 10 possible points for exercise 1,
  - 12 of 20 possible points for 2a + 2b,
  - 12 of 20 possible points for 3a + 3b.
- Important: Extensions will only be given in case of illness, and re-submissions will not be possible.
- See course page for the schedule.

Reading list

Obligatory reading; *selected parts* from:

- Jurafsky & Martin (2008):
  Speech and Language Processing (2nd Ed.)
- Seibel (2005):
  Practical Common Lisp (available online)
- Manning, Raghavan, & Schütze (2008):
  Introduction to Information Retrieval (available online)

Reading list

Obligatory reading; *selected parts* from:

- Jurafsky & Martin (2008):
  Speech and Language Processing (2nd Ed.)
- Seibel (2005):
  Practical Common Lisp (available online)
- Manning, Raghavan, & Schütze (2008):
  Introduction to Information Retrieval (available online)

Other recommended resources:

- Despite being 20 years old and long out-of-print *On Lisp* by Paul Graham is still a great read.
  - Freely available on-line: `http://www.paulgraham.com/onlisp.html`
- The Common Lisp 'HyperSpec':
  - `http://www.lispworks.com/documentation/HyperSpec/Front/`

Contact

- Questions?
  - Piazza: Online discussion board: `https://piazza.com/uio.no/fall2015/inf4820/`
  - `inf4820-help@ifi.uio.no` reaches all course staff:
  - Erik Velldal
  - Stephan Oepen
  - Siver Kjelberg Volle (laboratory assistant and grader)

Contact

- Questions?
  - Piazza: Online discussion board: `https://piazza.com/uio.no/fall2015/inf4820/`
  - `inf4820-help@ifi.uio.no` reaches all course staff:
  - Erik Velldal
  - Stephan Oepen
  - Siver Kjelberg Volle (laboratory assistant and grader)
- Messages:
  - Check your UiO email regularly;
  - Subscribe to the RSS message feed of the course page;
  - Participate on the online discussion board.

Overview of lectures

- Common Lisp basics
- Vector space models
- Classification and clustering
- Sequence models: n-grams and Hidden Markov Models
- Statistical parsing
- Recurring themes: Machine learning, scalable data representations, search, dynamic programming.