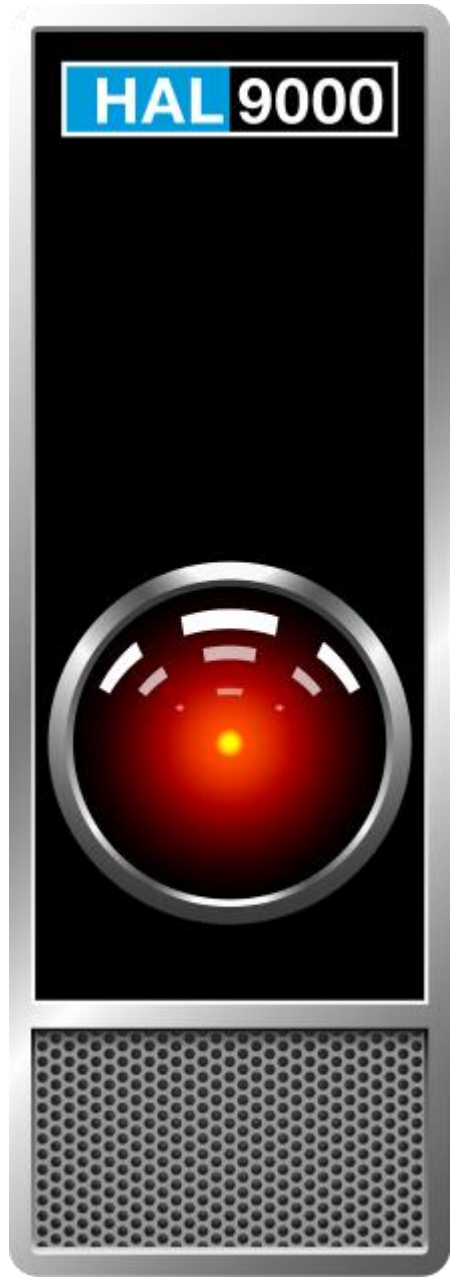

HAL 9000

2001: A Space Odyssey (1968) - A Conversation with HAL!

https://www.youtube.com/watch?v=r13I-TuDcWI

2001: A Space Odyssey (1968) - HAL Reads Lips

https://www.youtube.com/watch?v=XDO8OYnmkNY

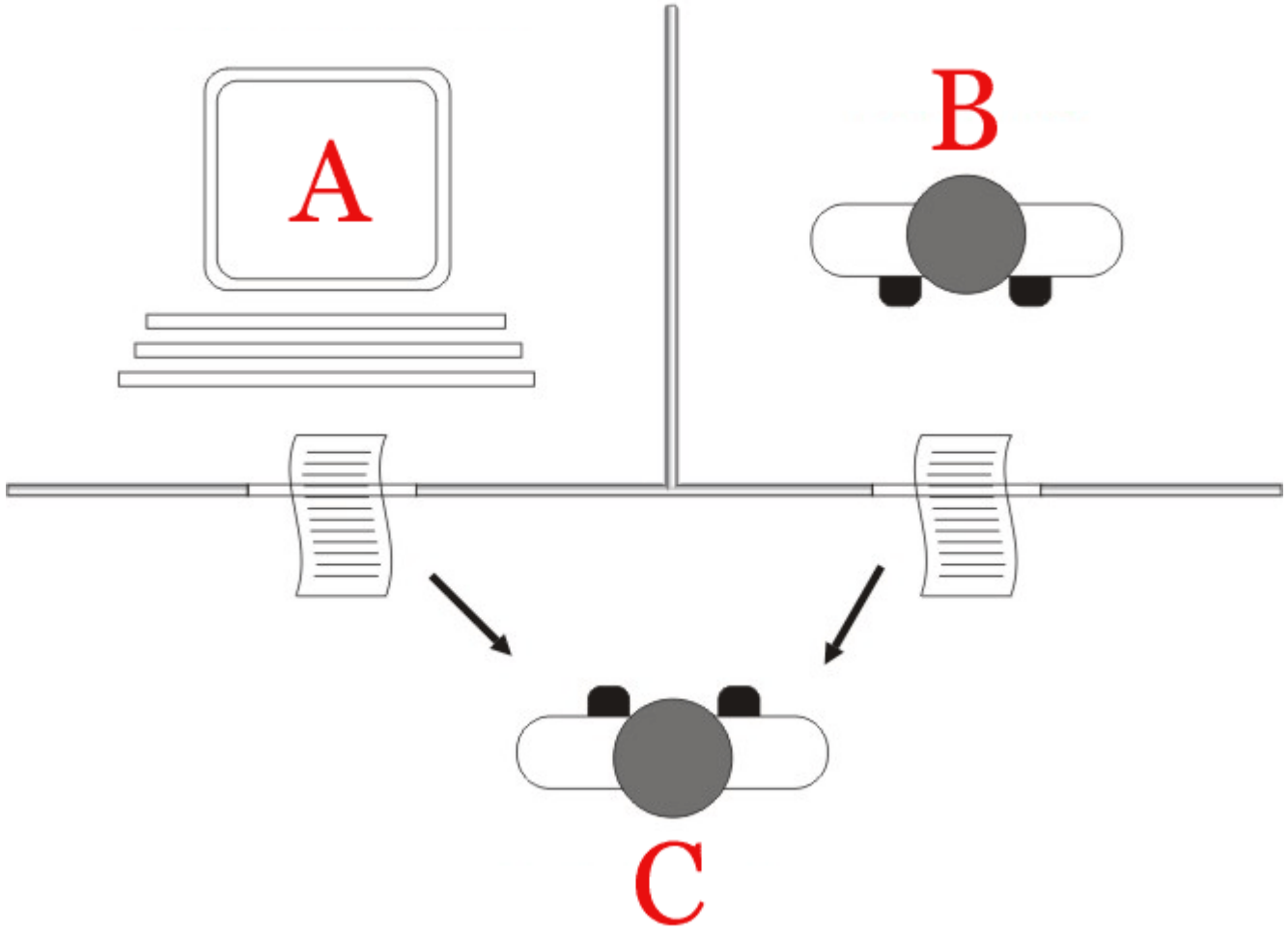
2001: A Space Odyssey (1968) – HAL: I'm Sorry, Dave!

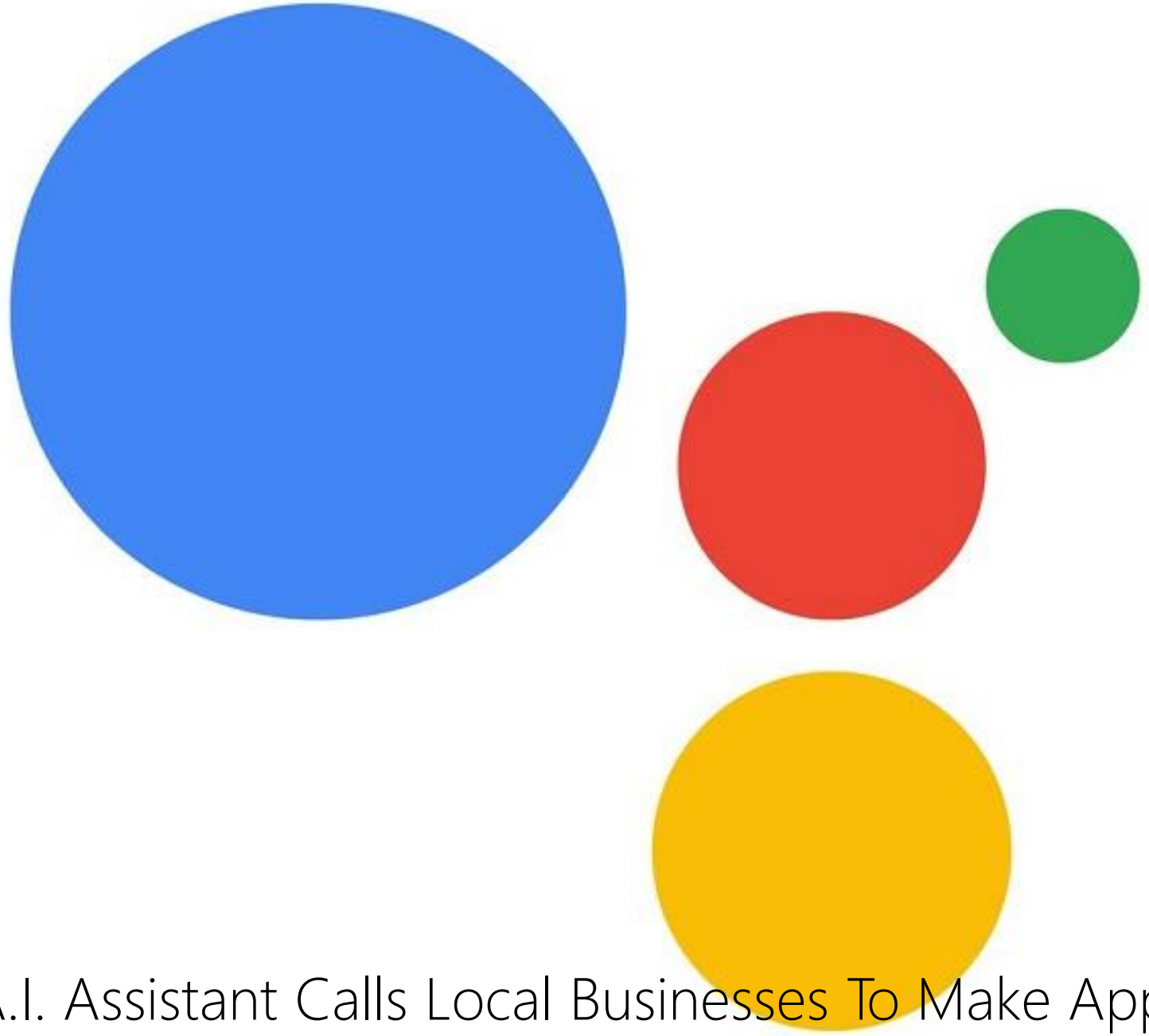
https://www.youtube.com/watch?v=Wy4EfdnMZ5g

The Turing Test

by Alan Turing in 1950

Player C, the interrogator, is given the task of trying to determine which player – A or B – is a computer and which is a human. The interrogator is limited to using the responses to written questions to make the determination.





Google Duplex: A.I. Assistant Calls Local Businesses To Make Appointments

https://www.youtube.com/watch?v=D5VN56jQMWM

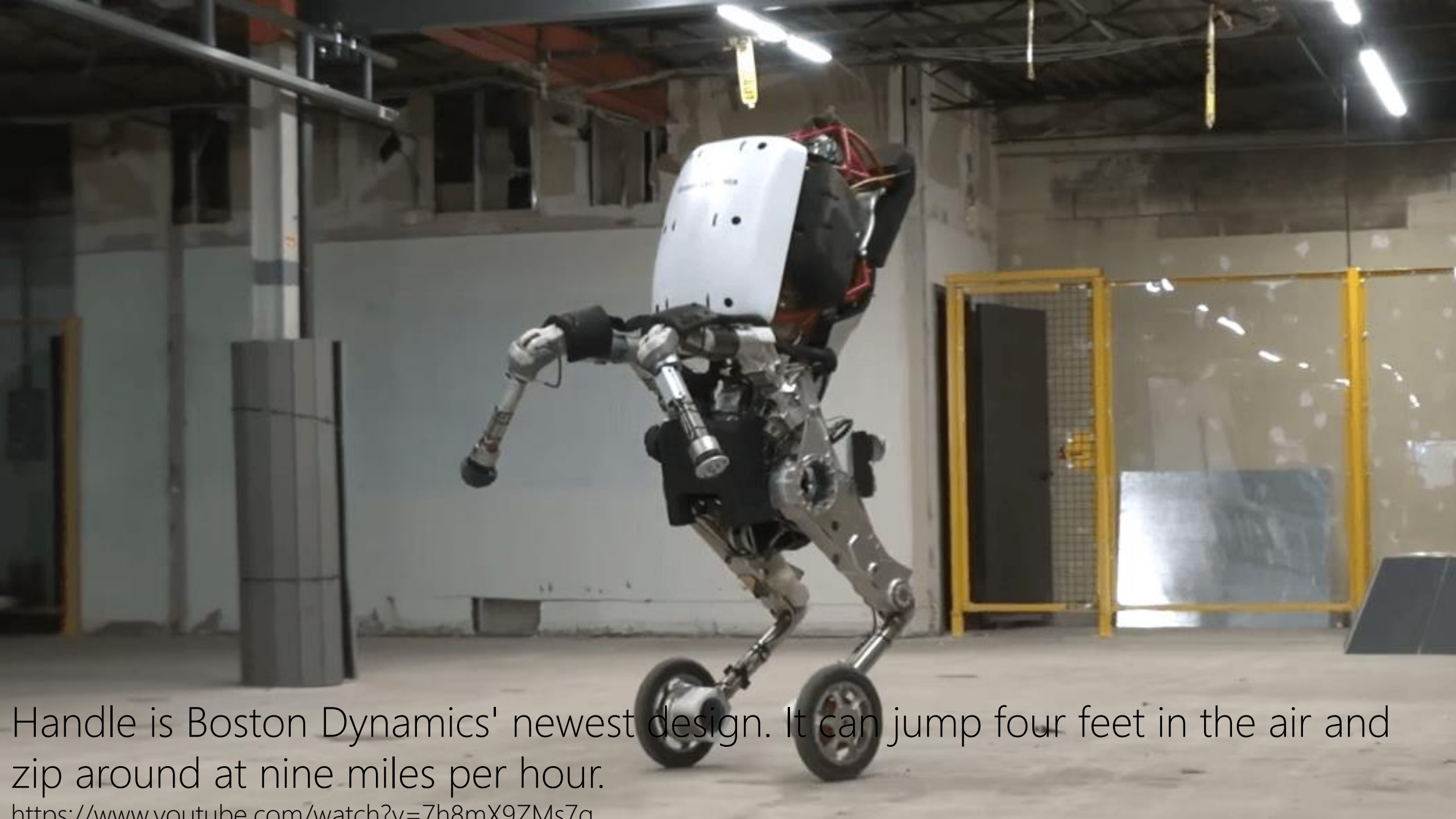
Natural Language Understanding
Speech, *Text*, Emotion, …

Research Priorities for Artificial Intelligence

The capacity for language is one of the central features of human intelligence and is therefore a prerequisite for artificial intelligence.

Despite its many practical applications, language is perhaps number 300 in the priority list for AI research. It would be a great achievement if AI could attain the capabilities of an orangutan, which do not include language!
- Yann LeCun (computer vision researcher)

Russell, Stuart, Daniel Dewey, and Max Tegmark. "Research priorities for robust and beneficial artificial intelligence." *Ai Magazine* 36.4 (2015): 105-114.



Handle is Boston Dynamics' newest design. It can jump four feet in the air and zip around at nine miles per hour.

https://www.youtube.com/watch?v=7h8mX97Ms7g

State of AI Report

October 1, 2020

Nathan Benaich Ian Hogarth

About the authors

Nathan Benaich

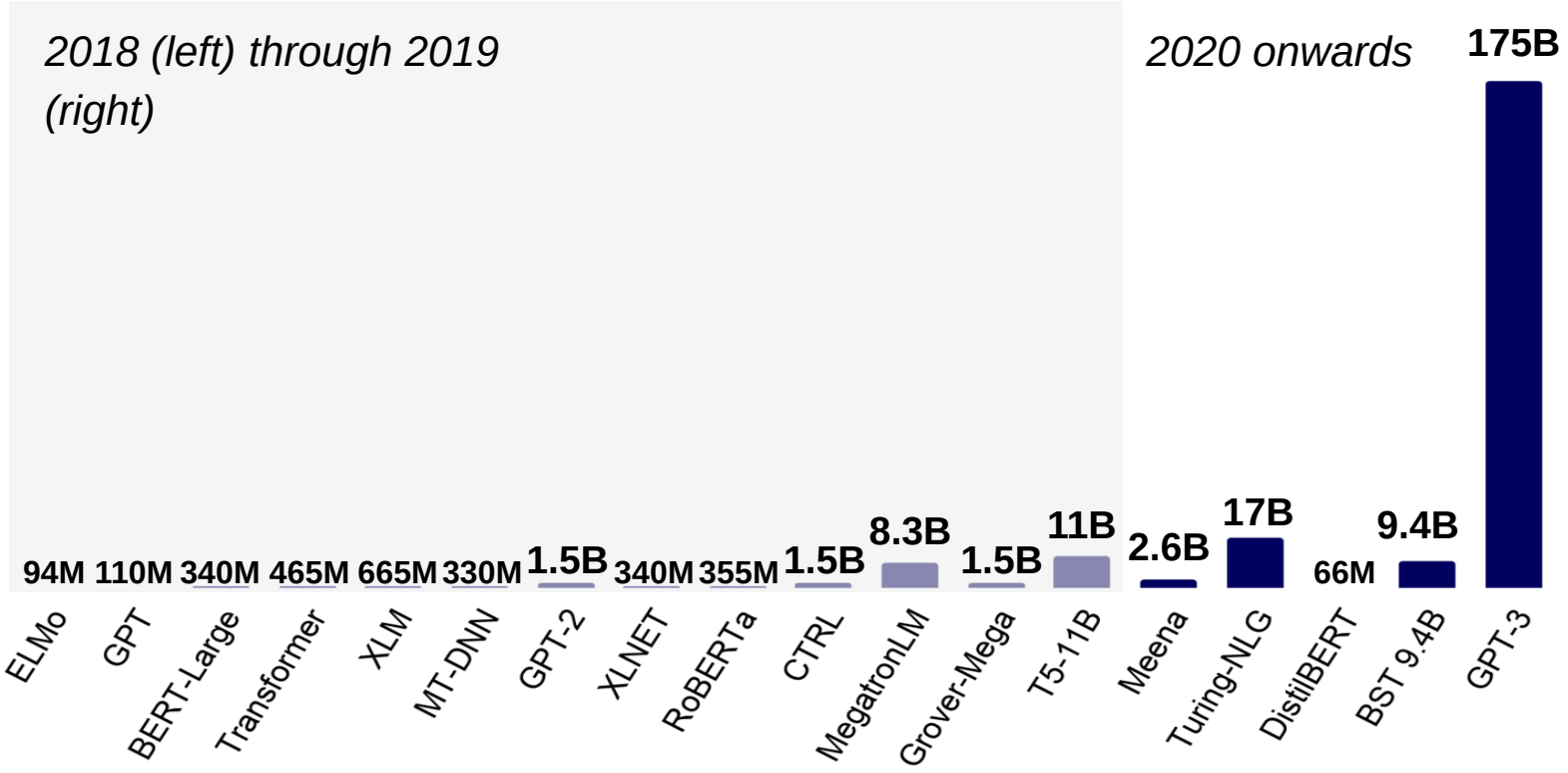
Nathan is the General Partner of **Air Street Capital**, a venture capital firm investing in AI-first technology and life science companies. He founded RAAIS and London.AI, which connect AI practitioners from large companies, startups and academia, and the RAAIS Foundation that funds open-source AI projects. He studied biology at Williams College and earned a PhD from Cambridge in cancer research.

Ian Hogarth

Ian is an **angel investor** in 60+ startups. He is a Visiting Professor at UCL working with Professor Mariana Mazzucato. Ian was co-founder and CEO of Songkick, the concert service used by 17M music fans each month. He studied engineering at Cambridge where his Masters project was a computer vision system to classify breast cancer biopsy images. He is the Chair of Phasecraft, a quantum software company.

Language models: Welcome to the Billion Parameter club

Huge models, large companies and massive training costs dominate the hottest area of AI today, NLP.

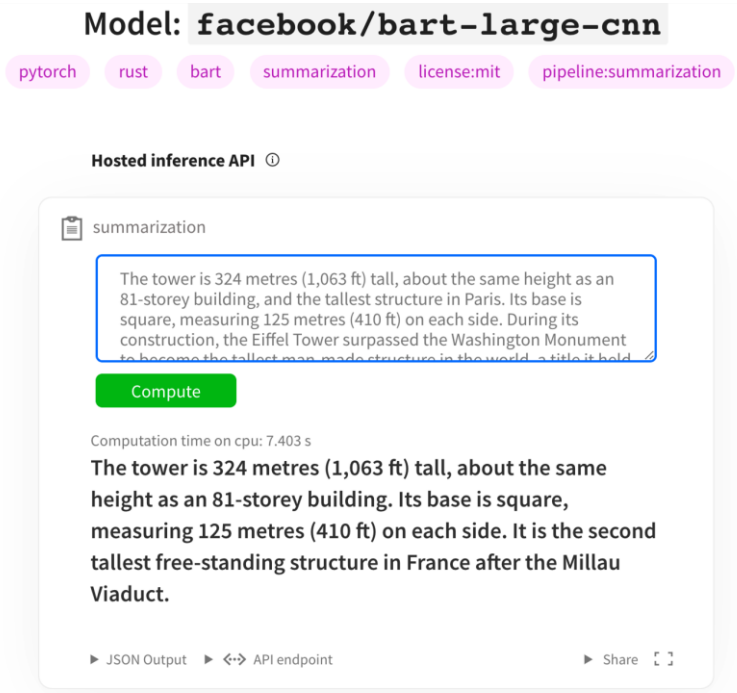


Note: The number of parameters indicates how many different coefficients the algorithm optimizes during the training process.

A new generation of transformer language models are unlocking new NLP use-cases

GPT-3, T5, BART are driving a drastic improvement in the performance of transformer models for text-to-text tasks like translation, summarization, text generation, text to code.

Summarization from huggingface.co/models



Code generation and more: gpt3examples.com

Computer, please convert my code into another programming language

An unsupervised machine translation model trained on GitHub projects with 1,000 parallel functions can translate 90% of these functions from C++ to Java and 57% of Python functions into C++ and successfully pass unit tests. No expert knowledge required, but no guarantees that the model didn't memorize the functions either.

Python input	Unsupervised C++ translation

```python
def SumOfKsubArray(arr, n, k):
  Sum = 0
  S = deque()
  G = deque()
  for i in range(k):
    while (len(S) > 0 and arr[S[-1]] >= arr[i]):
      S.pop()
    while (len(G) > 0 and arr[G[-1]] <= arr[i]):
      G.pop()
    G.append(i)
    S.append(i)
  for i in range(k, n):
    Sum += arr[S[0]] + arr[G[0]]
    while (len(S) > 0 and S[0] <= i - k):
      S.popleft()
    while (len(G) > 0 and G[0] <= i - k):
      G.popleft()
    while (len(S) > 0 and arr[S[-1]] >= arr[i]):
      S.pop()
    while (len(G) > 0 and arr[G[-1]] <= arr[i]):
      G.pop()
    G.append(i)
    S.append(i)
  Sum += arr[S[0]] + arr[G[0]]
  return Sum
```

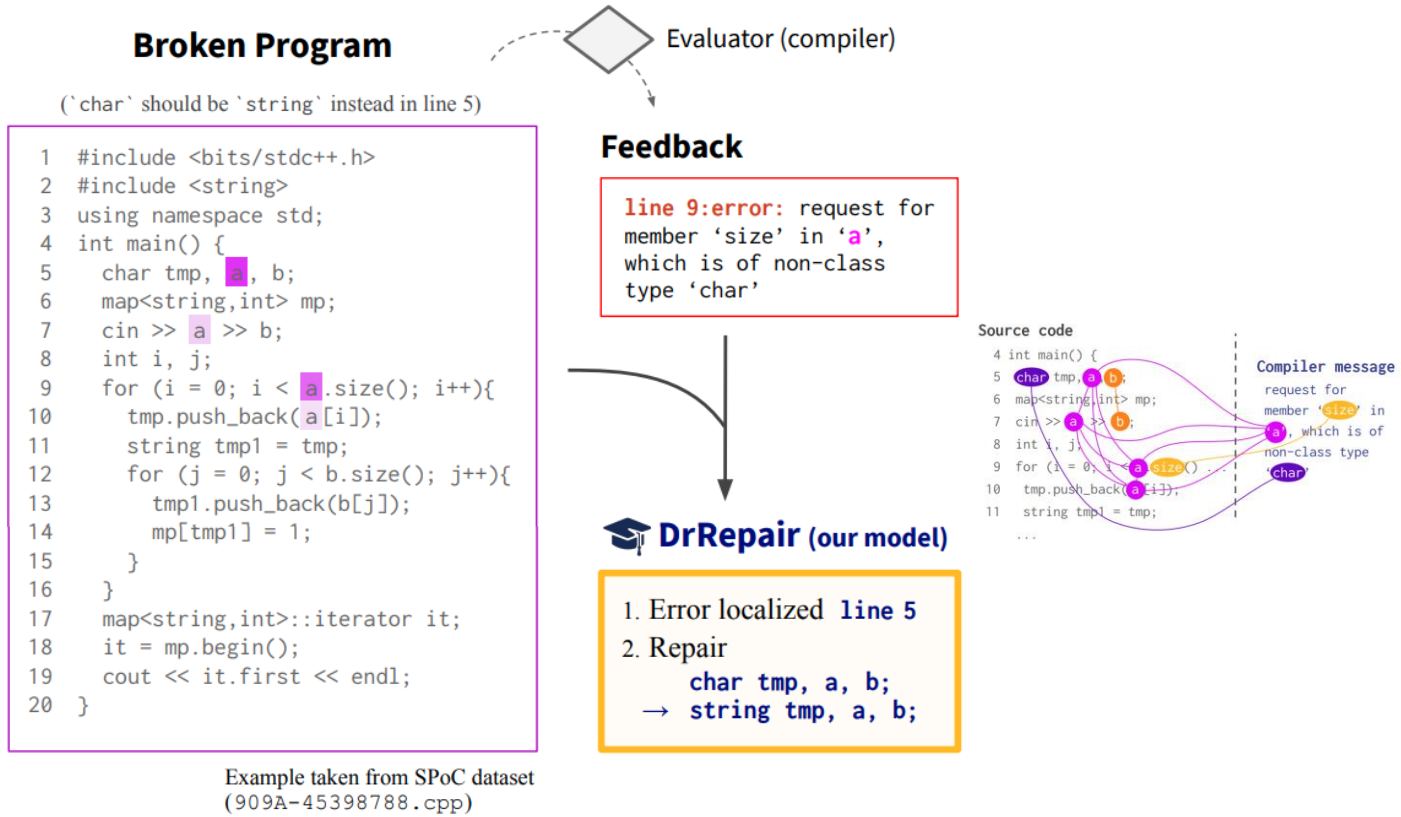
```cpp
int SumOfKsubArray(int arr[], int n, int k){
  int Sum = 0;
  deque <int> S;
  deque <int> G;
  for(int i = 0; i < k; i ++){
    while((int) S.size() > 0 && arr[S.back()] >= arr[i])
      S.pop_back();
    while((int) G.size() > 0 && arr[G.back()] <= arr[i])
      G.pop_back();
    G.push_back(i);
    S.push_back(i);
  }
  for(int i = k; i < n; i ++){
    Sum += arr[S.front()] + arr[G.front()];
    while((int) S.size() > 0 && S.front() <= i - k)
      S.pop_front();
    while((int) G.size() > 0 && G.front() <= i - k)
      G.pop_front();
    while((int) S.size() > 0 && arr[S.back()] >= arr[i])
      S.pop_back();
    while((int) G.size() > 0 && arr[G.back()] <= arr[i])
      G.pop_back();
    G.push_back(i);
    S.push_back(i);
  }
  Sum += arr[S.front()] + arr[G.front()];
  return Sum;
}
```

Figure 2: **Example of unsupervised Python to C++ translation**. TransCoder successfully translates the Python input function `SumOfKsubArray` into C++. TransCoder infers the types of the arguments, of the variables, and the return type of the function. The model maps the Python `deque()` container, to the C++ implementation `deque<>`, and uses the associated `front`, `back`, `pop_back` and `push_back` methods to retrieve and insert elements into the `deque`, instead of the Python square brackets `[]`, `pop` and `append` methods. Moreover, it converts the Python `for` loop and `range` function properly.

Computer, can you automatically repair my buggy programs too?

Given a broken program and diagnostic feedback (compiler error message), DrRepair localizes an erroneous line and generates a repaired line.

- The model jointly reasons over the broken source code and the diagnostic feedback using graph neural networks.
- They use self-supervised learning to obviate the need for labelling by taking code from programming competitions and corrupting it into a broken program.
- A SOTA is set on DeepFix, which is a program repair benchmark for correct intro programming assignments in C.



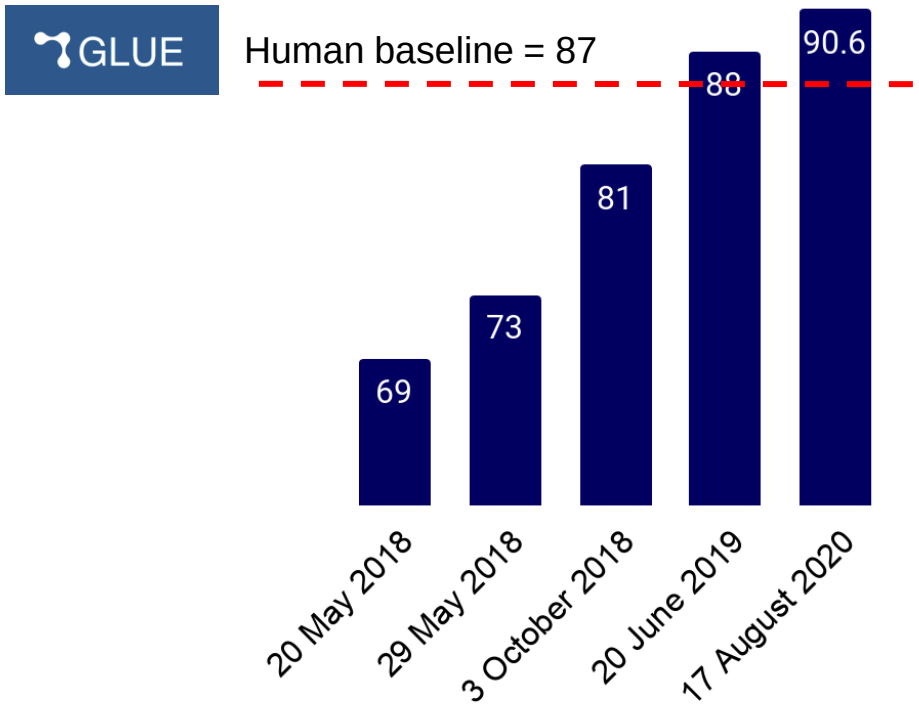
Example taken from SPoC dataset (909A-45398788.cpp)

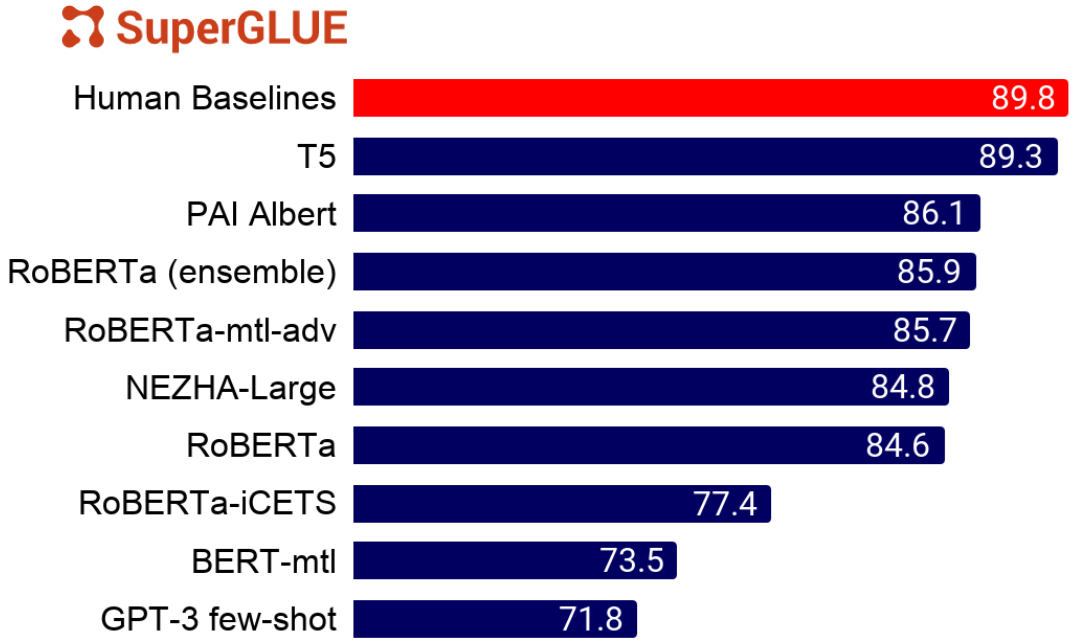
Stanford University

NLP benchmarks take a beating: Over a dozen teams outrank the human GLUE baseline

It was only 12 months ago that the human GLUE benchmark was beat by 1 point. Now SuperGLUE is in sight.

- GLUE and it's more challenging sibling SuperGLUE are benchmarks that evaluate NLP systems at a range of tasks spanning logic, common sense understanding, and lexical semantics. The human benchmark on GLUE is reliably beat today (right) and the SuperGLUE human benchmark is almost surpassed too!





Tuning billions of model parameters costs millions of dollars

Based on variables released by Google et al., you're paying circa $1 per 1,000 parameters. This means OpenAI's 175B parameter GPT-3 could have cost tens of millions to train. Experts suggest the likely budget was $10M.

Just how much does it cost to train a model? Two correct answers are "depends" and "a lot". More quantitatively, here are current ballpark list-price costs of training differently sized BERT [4] models on the Wikipedia and Book corpora (15 GB). For each setting we report two numbers - the cost of one training run, and a typical fully-loaded cost (see discussion of "hidden costs" below) with hyper-parameter tuning and multiple runs per setting (here we look at a somewhat modest upper bound of two configurations and ten runs per configuration).[4]

- $2.5k - $50k (110 million parameter model)
- $10k - $200k (340 million parameter model)
- $80k - $1.6m (1.5 billion parameter model)

For example, based on information released by Google, we estimate that, at list-price, training the 11B-parameter variant[5] of T5 [5] cost well above $1.3 million for a single run. Assuming 2-3 runs of the large model and hundreds of the small ones, the (list-)price tag for the entire project may have been $10 million[6].

Not many companies – certainly not many startups – can afford this cost. Some argue that this is not a severe issue; let the Googles of the world pre-train and publish the large language models, and let the rest of the world fine-tune them (a much cheaper endeavor) to specific tasks. Others (e.g., Etchemendy and Li [6]) are not as sanguine.

Artificial intelligence / Machine learning

Training a single AI model can emit as much carbon as five cars in their lifetimes

Deep learning has a terrible carbon footprint.

by **Karen Hao**

June 6, 2019

Common carbon footprint benchmarks

in lbs of CO2 equivalent

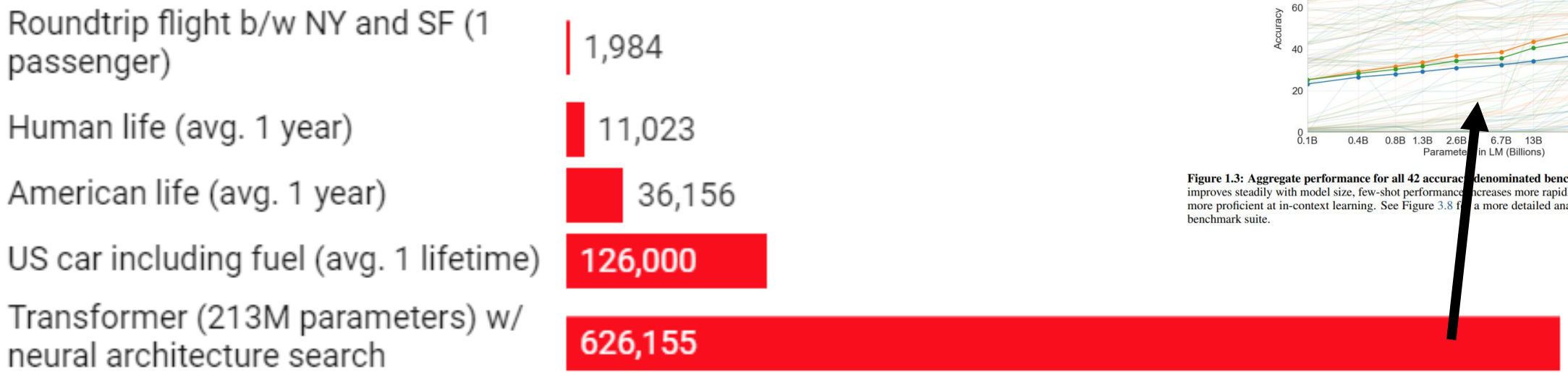


Chart: MIT Technology Review • Source: Strubell et al. • Created with Datawrapper

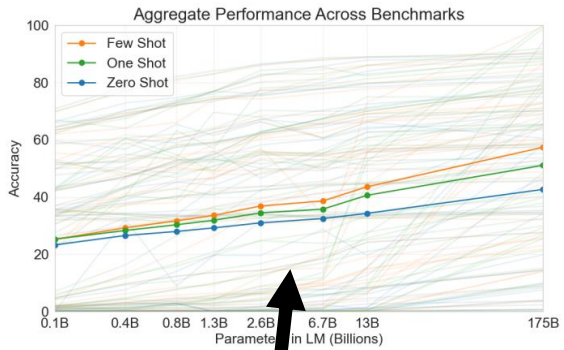


Figure 1.3: Aggregate performance for all 42 accuracy-denominated benchmarks While zero-shot performance improves steadily with model size, few-shot performance increases more rapidly, demonstrating that larger models are more proficient at in-context learning. See Figure 3.8 for a more detailed analysis on SuperGLUE, a standard NLP benchmark suite.

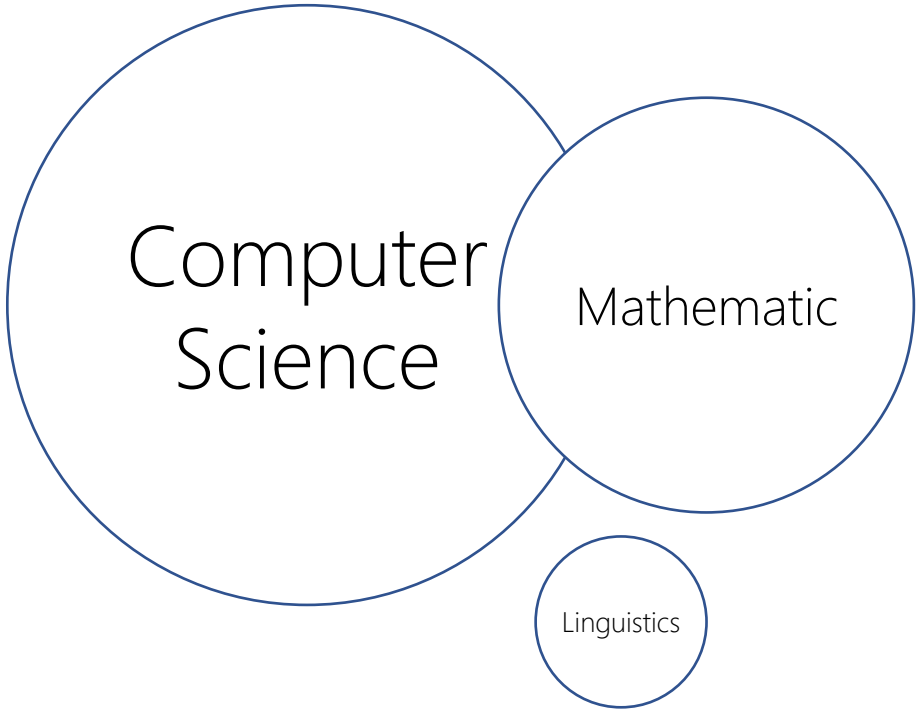
Natural Language Processing & Understanding

COMP8730 Winter 2021

Image Source: https://unanimous.ai/chat-with-a-different-kind-of-artificial-intelligence/

BACKGROUND

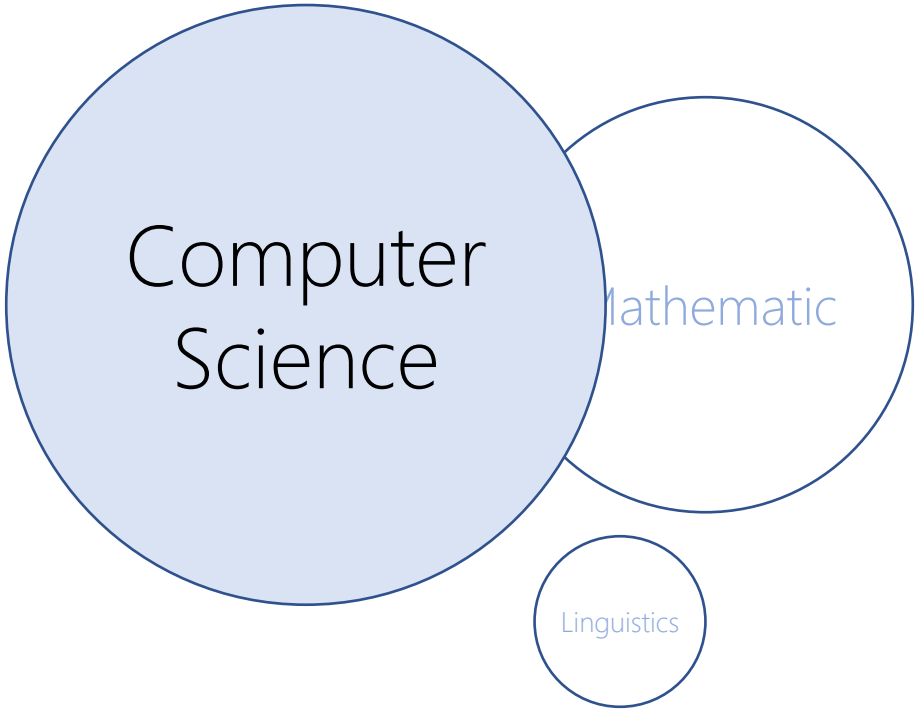
Background



Background: *the course is targeted at computer scientists!*

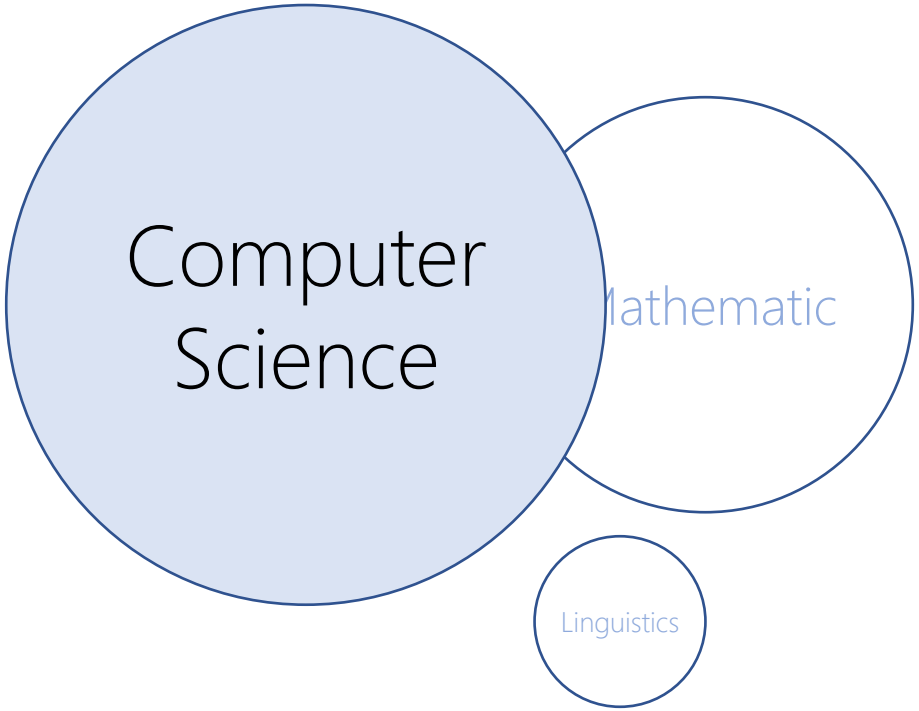
Assumed to know

- Design of Algorithms
  - Greedy
  - Dynamic Programming
  - Divide-Conquer
  - Recursion
  - Backtracking
- Analysis of Algorithms
  - Time & Space (memory)
  - Big O
  - Complexity Theory
- Data Modeling
  - Data Structure



Background: *the course is targeted at computer scientists!*

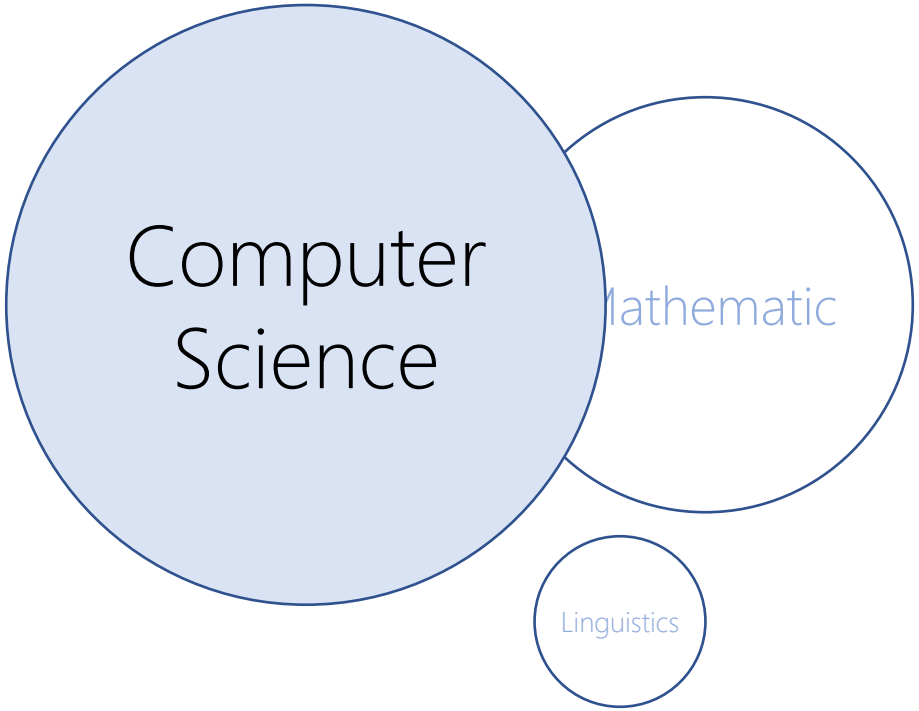
Courses
- COMP**2**540: Data Structures and Algorithms
- COMP**3**540: Theory of Computation
- COMP**4**540: Design and Analysis of Computer Algorithms



Background: *the course is targeted at computer scientists!*

References

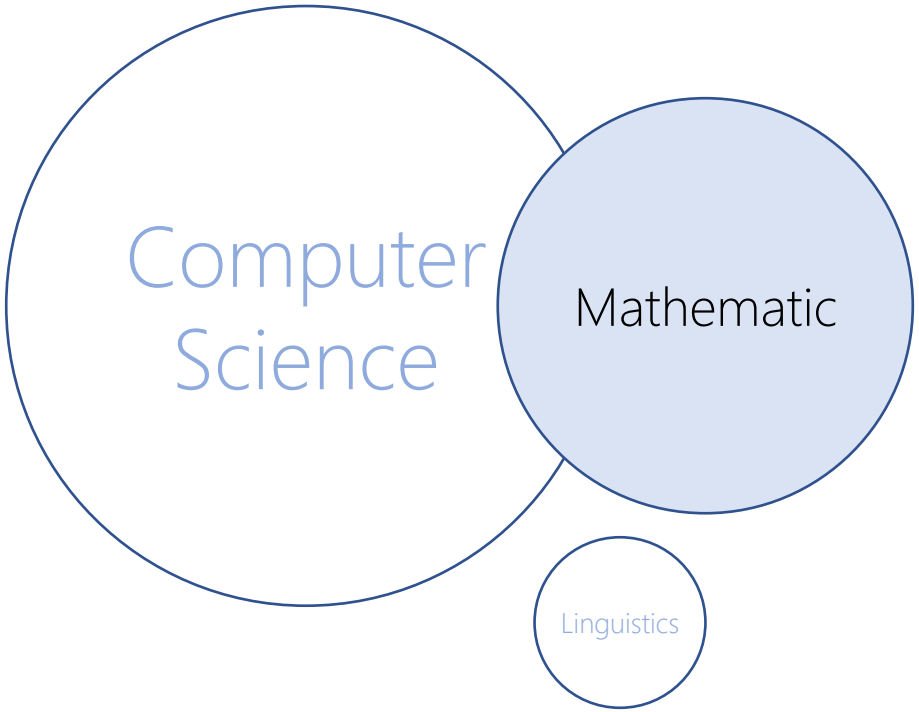
- Introduction to algorithms (3rd ed.), Cormen, et al. (2009).
- Introduction to the Theory of Computation. Sipser, M. (2012).



Background: *the course is targeted at computer scientists!*

Assumed to know

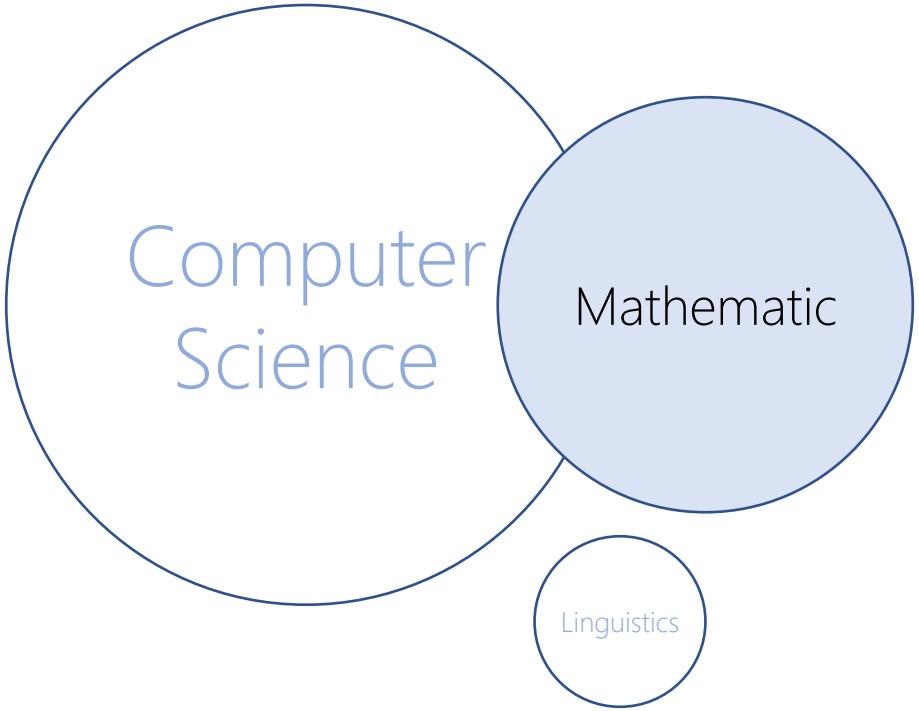
- Multivariate Calculus
  - Derivatives
  - Partial Derivatives
- Linear Algebra
  - Vectors
  - Matrices
- Probability & Statistics



Background: *the course is targeted at computer scientists!*

References

- Mathematics for Machine Learning, Deisenroth et al.
a draft can be found online at https://mml-book.github.io/



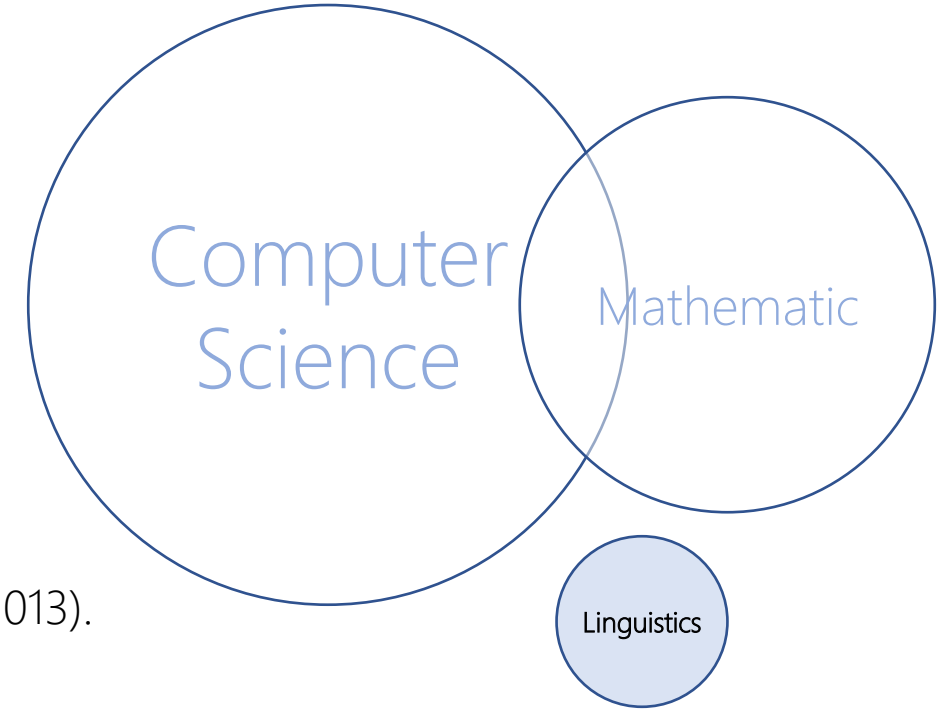
Background: *the course is targeted at computer scientists!*

Assumed to know

- Elementary concepts
  - Part-of-Speech (Nouns, Verbs, …)
  - English Grammar

References

- Linguistic Fundamentals for Natural Language Processing:
100 Essentials from Morphology and Syntax, Bender, E. M. (2013).



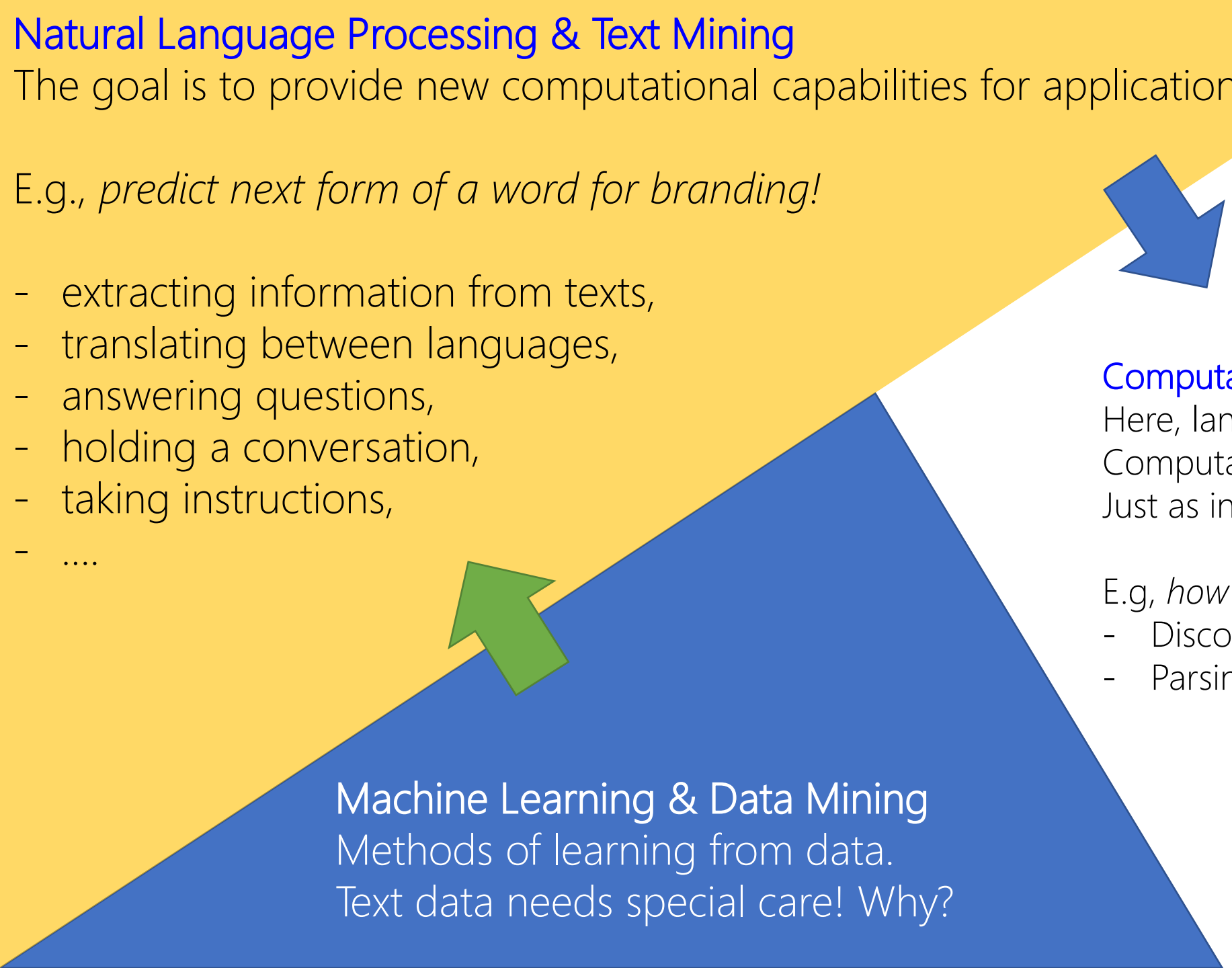
APPROACH

Natural Language Processing & Text Mining

The goal is to provide new computational capabilities for applications

E.g., *predict next form of a word for branding!*

- extracting information from texts,
- translating between languages,
- answering questions,
- holding a conversation,
- taking instructions,
- ....



Computational Linguistics

Here, language is the object of study.
Computational methods are to support.
Just as in computational biology

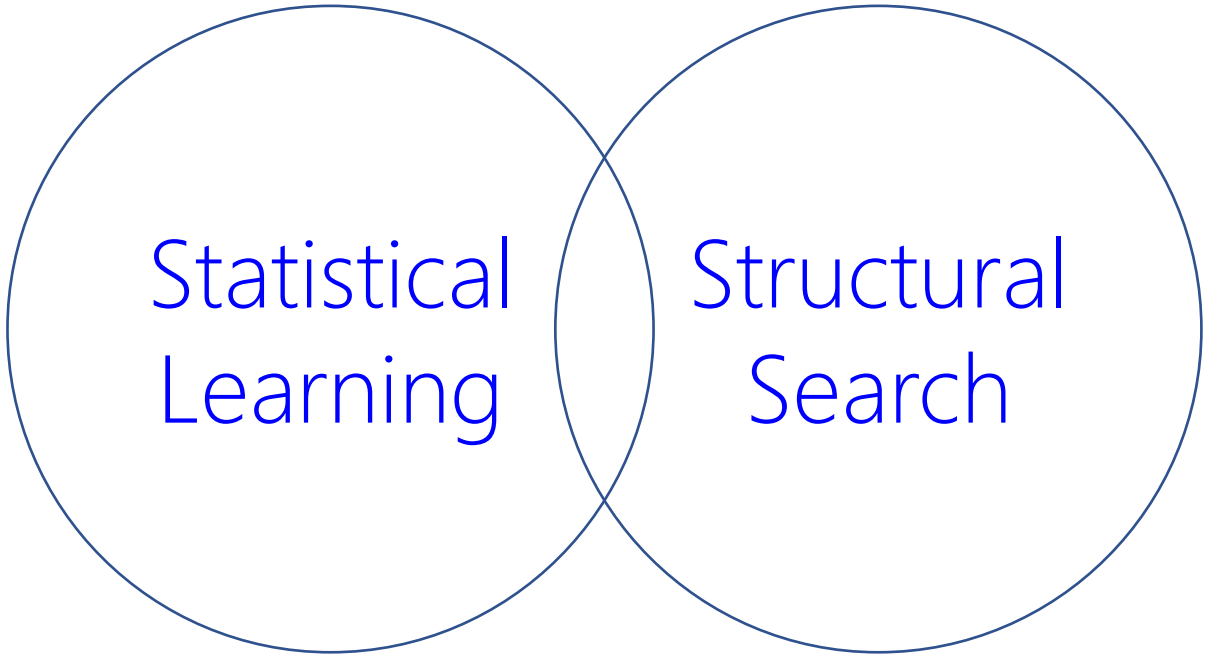
E.g, *how a word is evolving in time?*

- Discourse analysis
- Parsing

Machine Learning & Data Mining

Methods of learning from data.
Text data needs special care! Why?

Approach: *unstructured vs. structured*



Approach: *unstructured vs. structured*

Examples

- Lemmatization

  converts the word to its meaningful base form, which is called Lemma. The same word may have multiple different Lemmas

- Stemming

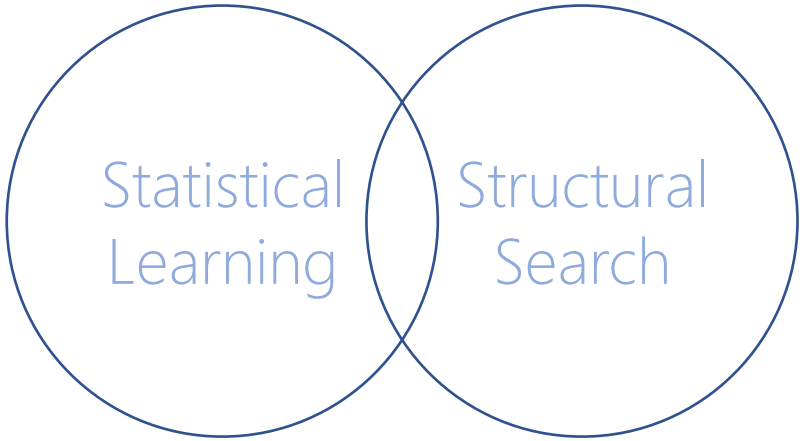
  removes or stems the last few characters of a word, often leading to incorrect meanings and spelling

- Segmentation

  dividing written text into meaningful units, such as words, sentences, or topics

- Tenses of a Verb

  present, present perfect, past tenses of verbs

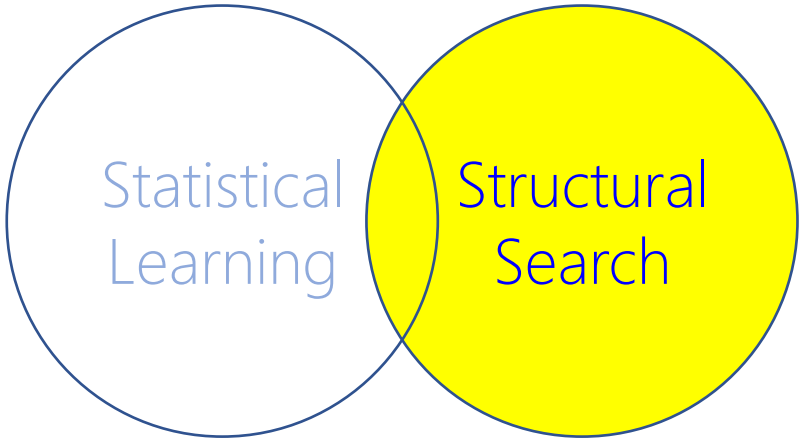


Approach: *unstructured vs. structured*

Examples

- Lemmatization → *lemma*(word) = drop '*s*' or '*es*' from the end

  converts the word to its meaningful base form, which is called Lemma. The same word may have multiple different Lemmas

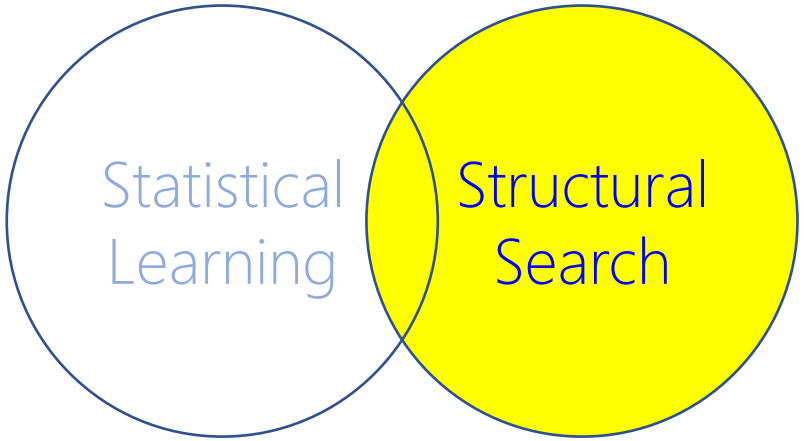


Approach: *unstructured vs. structured*

Examples

- Lemmatization → *lemma*(‘*books*’) = ‘*book*’

  converts the word to its meaningful base form, which is called Lemma. The same word may have multiple different Lemmas

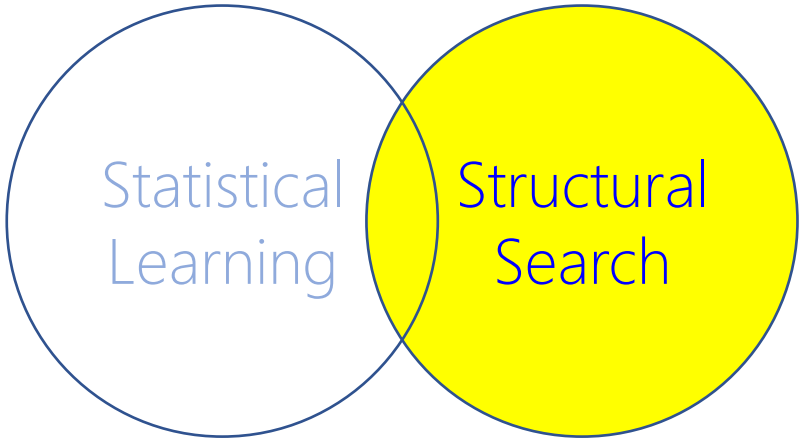


Approach: *unstructured vs. structured*

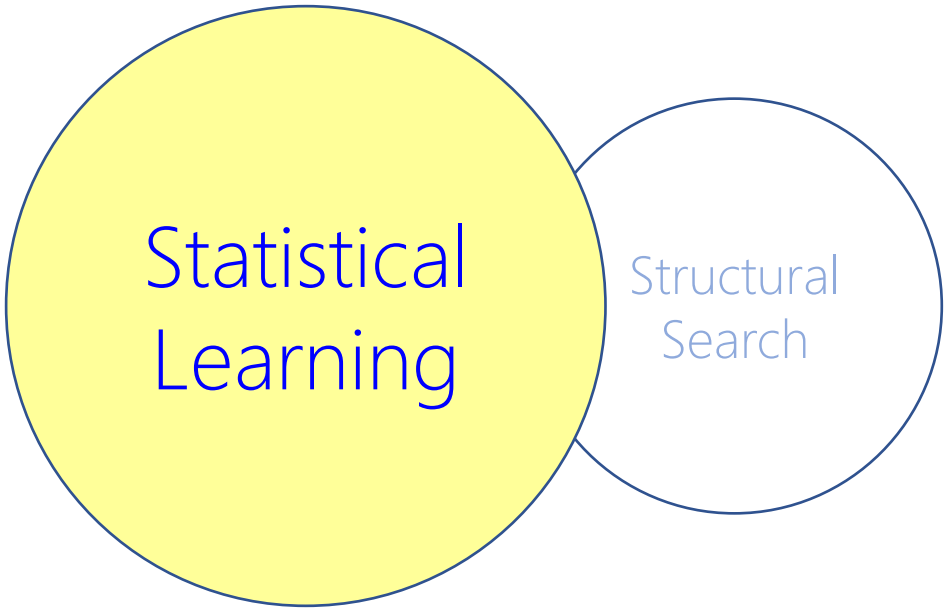
Examples

- Lemmatization → *lemma*(‘*indices*’) = ‘*indic*’

  converts the word to its meaningful base form, which is called Lemma. The same word may have multiple different Lemmas



Approach: *unstructured vs. structured*



Approach: *unstructured vs. structured*

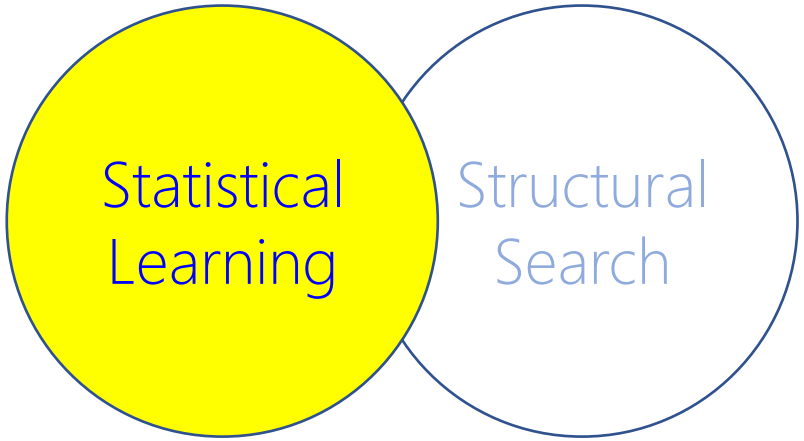
Examples

- Lemmatization → *lemma*(‘*indices*’)

  converts the word to its meaningful base form, which is called Lemma. The same word may have multiple different Lemmas

https://en.wikipedia.org/wiki/Database_index

A database index is a data structure that improves the speed of data retrieval operations on a database table at the cost of additional writes and storage space to maintain the index data structure. Indexes are used to quickly locate data without having to search every row in a database table every time a database table is accessed. Indexes can be created using one or more columns of a database table, providing the basis for both rapid random lookups and efficient access of ordered records. An index is a copy of selected columns of data from a table, called a database key or simply key, that can be searched very efficiently that also includes a low-level disk block address or direct link to the complete row of data it was copied from. Some databases extend the power of indexing by letting developers create indexes on functions or expressions. For example, an index could be created on upper(last_name), which would only store the upper-case versions of the last_name field in the index. Another option sometimes supported is the use of partial indices, where index entries are created only for those records that satisfy some conditional expression. A further aspect of flexibility is to permit indexing on user-defined functions, as well as expressions formed from an assortment of built-in functions.



Approach: *unstructured vs. structured*

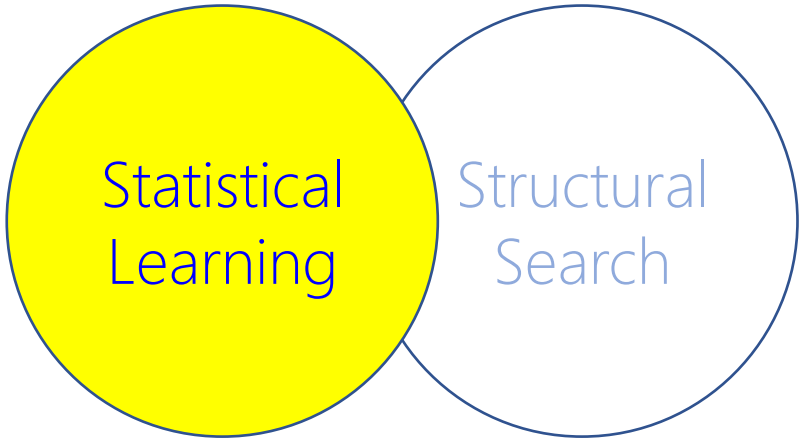
Examples

- Lemmatization → *lemma*(‘*indices*’)

  converts the word to its meaningful base form, which is called Lemma. The same word may have multiple different Lemmas

https://en.wikipedia.org/wiki/Database_index

A database index is a data structure that improves the speed of data retrieval operations on a database table at the cost of additional writes and storage space to maintain the index data structure. Indexes are used to quickly locate data without having to search every row in a database table every time a database table is accessed. Indexes can be created using one or more columns of a database table, providing the basis for both rapid random lookups and efficient access of ordered records. An index is a copy of selected columns of data from a table, called a database key or simply key, that can be searched very efficiently that also includes a low-level disk block address or direct link to the complete row of data it was copied from. Some databases extend the power of indexing by letting developers create indexes on functions or expressions. For example, an index could be created on upper(last_name), which would only store the upper-case versions of the last_name field in the index. Another option sometimes supported is the use of partial indices, where index entries are created only for those records that satisfy some conditional expression. A further aspect of flexibility is to permit indexing on user-defined functions, as well as expressions formed from an assortment of built-in functions.



Approach: *unstructured vs. structured*

Examples

- Segmentation

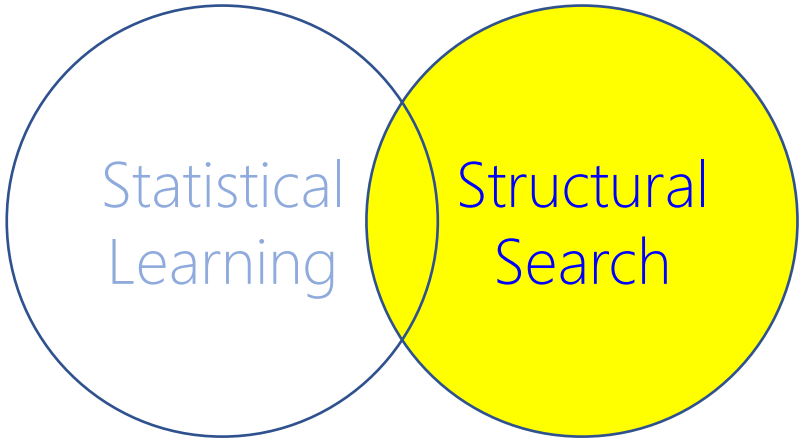
  dividing written text into meaningful units, such as words (tokenizer), sentences, or topics

https://www.nltk.org/_modules/nltk/tokenize/punkt.html

Punkt knows that the periods in Mr. Smith and Johann S. Bach do not mark sentence boundaries. And sometimes sentences can start with non-capitalized words. i is a good variable name.

https://en.wikipedia.org/wiki/Database_index

Adatabaseindexisadatastructurethatimprovesthespeedofdataretrievaloperationsonadatabasetable atthecostofadditionalwritesandstoragespacetomaintaintheindexdatastructure.Indexesareusedtoqui cklylocatedatawithouthavingtosearcheveryrowinadatabasetableeverytimeadatabasetableisaccesse d.Indexescanbecreatedusingoneormorecolumnsofadatabasetableasexplainedin Sect.2.1.4,providin gthebasisforbothrapidrandomlookupsandefficientaccessoforderedrecords.Anindexisacopyofselect edcolumnsofdatafromatable,calledadatabasekeyorsimplykey,thatcanbesearchedveryefficientlythat alsoincludesalowleveldiskblockaddressordirectlinktothecompleterowofdataitwascopiedfrom.Some databasesextendthepowerofindexingbylettingdeveloperscreateindexesonfunctionsorexpressions.F orexample,anindexcouldbecreatedonupper(last_name),whichwouldonlystoretheuppercaseversions ofthelast_namefieldintheindex.Anotheroptionsometimessupportedistheuseofpartialindices,wherein dexentriesarecreatedonlyforthoserecordsthatsatisfysomeconditionalexpression.Afurtheraspectoffle xibilityistopermitindexingonuserdefinedfunctions,aswellasexpressionsformedfromanassortmentofb uilt-infunctions.



Approach: *unstructured vs. structured*

Examples

- Segmentation

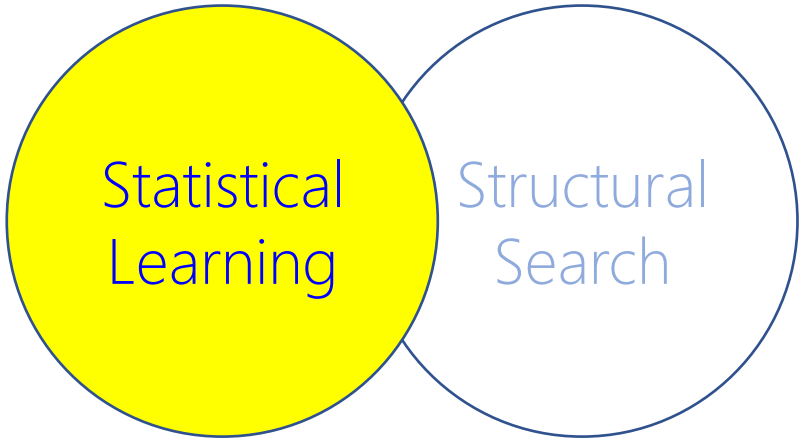
  dividing written text into meaningful units, such as words (tokenizer), sentences, or topics

https://www.nltk.org/_modules/nltk/tokenize/punkt.html

Punkt knows that the periods in Mr. Smith and Johann S. Bach do not mark sentence boundaries. And sometimes sentences can start with non-capitalized words. i is a good variable name.

https://en.wikipedia.org/wiki/Database_index

Adatabaseindexisadatastructurethatimprovesthespeedofdataretrievaloperationsonadatabasetable atthecostofadditionalwritesandstoragespacetomaintaintheindexdatastructure.Indexesareusedtoqui cklylocatedatawithouthavingtosearcheveryrowinadatabasetableeverytimeadatabasetableisaccesse d.Indexescanbecreatedusingoneormorecolumnsofadatabasetableasexplainedin Sect.2.1.4,providin gthebasisforbothrapidrandomlookupsandefficientaccessoforderedrecords.Anindexisacopyofselect edcolumnsofdatafromatable,calledadatabasekeyorsimplykey,thatcanbesearchedveryefficientlythat alsoincludesalowleveldiskblockaddressordirectlinktothecompleterowofdataitwascopiedfrom.Some databasesextendthepowerofindexingbylettingdeveloperscreateindexesonfunctionsorexpressions.F orexample,anindexcouldbecreatedonupper(last_name),whichwouldonlystoretheuppercaseversions ofthelast_namefieldintheindex.Anotheroptionsometimessupportedistheuseofpartialindices,wherein dexentriesarecreatedonlyforthoserecordsthatsatisfysomeconditionalexpression.Afurtheraspectoffle xibilityistopermitindexingonuserdefinedfunctions,aswellasexpressionsformedfromanassortmentofb uilt-infunctions.



The debate about the relative importance of machine learning and linguistic knowledge sometimes becomes heated. No machine learning specialist likes to be told that their engineering methodology is unscientific alchemy;[5] nor does a linguist want to hear that the search for general linguistic principles and structures has been made irrelevant by big data. Yet there is clearly room for both types of research: we need to know how far we can go with end-to-end learning alone, while at the same time, we continue the search for linguistic representations that generalize across applications, scenarios, and languages. For more on the history of this debate, see Church (2011); for an optimistic view of the potential symbiosis between computational linguistics and deep learning, see Manning (2015).

Manning, C. D. (2015). Last words: Computational linguistics and deep learning. Computational Linguistics 41(4), 701–707.

NLP is AI-hard (AI-Complete)

- Fundamentally Discrete

- Compositional

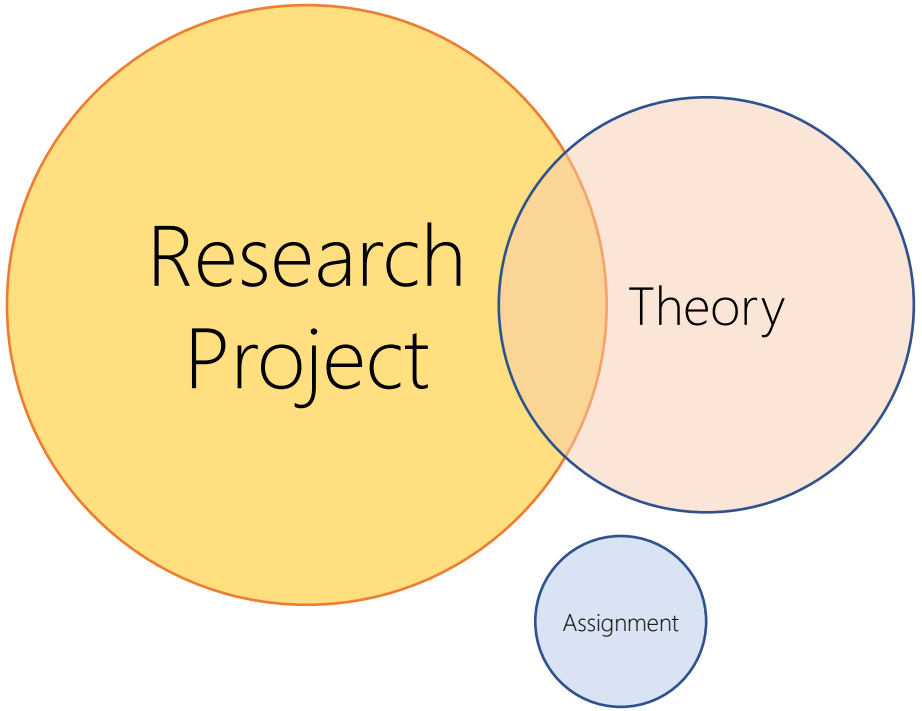
  meaning created by specific combinatorial arrangements of units (chars, words, sentences)

- Distribution over words is power law

  there will be a few words that are very frequent
  long tail of words that are rare
  consequently, NLP algorithms must be especially robust to observations that do not occur in the training data

COURSE

Course: Research-oriented, Project-driven



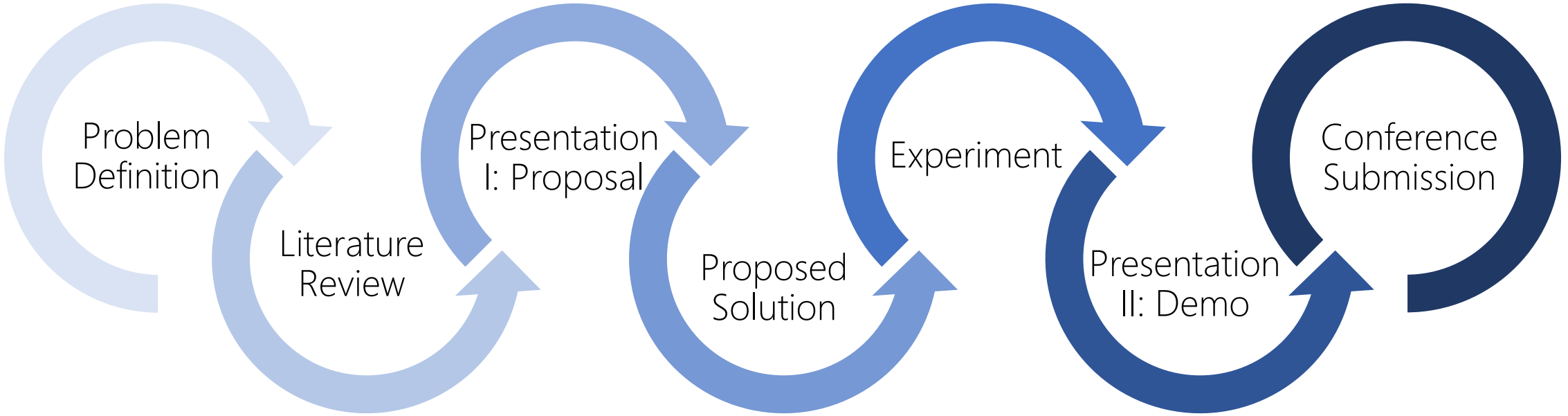
RESEARCH PROJECT

RESEARCH PROJECT

VS.

SOFTWARE PROJECT

Research Project



***Research* Project**	**Problem Definition (05%)**	Authors (Team Members, Max: 2 Students) Abstract Motivation (Application?) Formal Definition (Mathematical Formulation)
	Literature Review (10%)	History [-∞, 2010] State of the Art (SOTA) [2011, 2021] Critics & Gaps (Regardless the Project Fills Them) What Gap(s) the Project Fills
	Presentation I: Proposal (05%)	10' + 5' Q&A
	Proposed Solution (10%)	Formal Definition (Mathematical Formulation) Implementation (Online Repo, preferably Github)
	Experiment (20%)	Datasets Evaluation Methodology Metrics Baselines Results
	Presentation II: Demo (05%)	10' + 5' Q&A
	Conference Submission (10%)	Pending Instructor's Approval - KES2021 - RecSys21 - [more]
	Conference Acceptance (Bonus)	Conference Registration Conference Travel

THEORY

Theoretical[2]	**Lec01B: Meet and Greet:** Course Outline	Jan. 07
	Lec02: Text Normalization	Jan. 12, 14
	Lec03: Text Vector Representation	Jan. 19, 21
	Lec04: Neural Text Vector Representation	Jan. 26, 28
	Lec05: Language Models	Feb. 02, 04
	Lec06: Neural Language Models	Feb. 09, 11
	Lec07: Reading Week: No Class	Feb. 13 – 21
	Lec08: Project Presentations	Feb. 23, 25
	Lec09: Learning w/ Supervision: Sentiment, Sense Disambiguation	Mar. 02, 04
	Lec10: Learning w/o Supervision I: Topic Modeling	Mar. 09, 11
	Lec11: Learning w/o Supervision II: User Modeling	Mar. 16, 18
	Lec12: Learning w/o Supervision III: Web Search	Mar. 23, 25
	Lec13: Project Presentations	Mar. 30, Apr. 01
	Lec14A: Peer Review	Apr. 06
	Lec15: Final Exam	Apr. 12 – 22

BOOKS & RESOURCES

Books	Introduction to Natural Language Processing Jacob Eisenstein ISBN: 9780262042840 536 pages October 2019 http://cseweb.ucsd.edu/~nnakashole/teaching/eisenstein-nov18.pdf Natural Language Processing for Social Media, Third Edition *Synthesis Lectures on Human Language Technologies* Anna Atefeh Farzindar, Diana Inkpen ISBN: 9781681738116 | PDF ISBN: 9781681738123 Hardcover ISBN: 9781681738130 Copyright © 2020 | 219 Pages DOI: 10.2200/S00999ED3V01Y202003HLT046 Speech and Language Processing, 3rd Edition Draft An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition Dan Jurafsky and James H. Martin Free Access → https://web.stanford.edu/~jurafsky/slp3/
Hands-on Resources	NLP with Python – Analyzing Text with the Natural Language Toolkit https://www.nltk.org/book/ NLP with PyTorch: Build Intelligent Language Applications Using Deep Learning. https://github.com/joosthub/PyTorchNLPBook

MARKING SCHEMA

Marking Scheme	***Research* Project** - Problem Definition - Literature Review - Presentation I: Proposal - Proposed Solution - Experiment - Presentation II: Demonstration - Conference Submission (Pending Instr. 's Approval)	65% - 05% - 10% - 05% - 10% - 20% - 05% - 10%
	Assignments (+peer review)	15%
	Midterm Exam	10%
	Final Exam	10%
Remarks	The written reports will be assessed not only on their academic merit but also on the student's communication skills as exhibited through the reports. To achieve a passing grade, the students must achieve at least 70% of the entire marking scheme. The fraction mark is rounded to the ceiling. The students earn final course grades as per the Senate policy for Grading and Calculation of Averages.	

ME

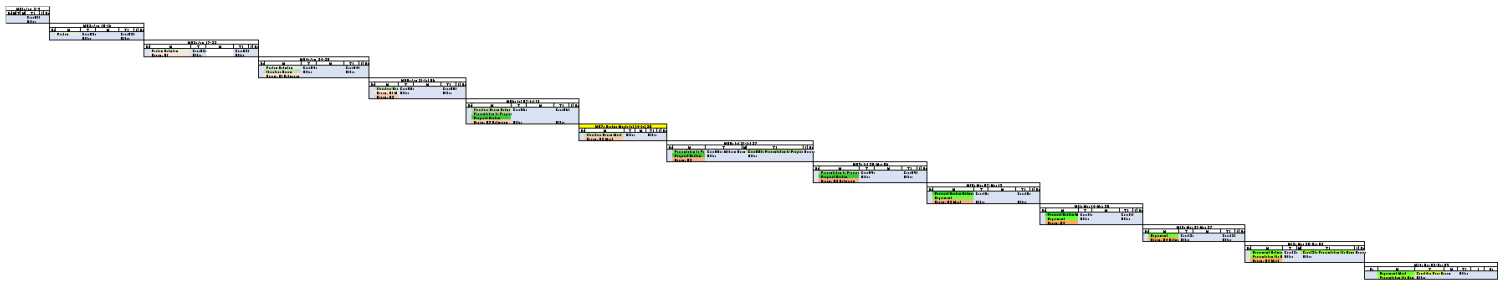
Hossein Fani, PhD
Assistant Professor, School of Computer Science, Faculty of Science, University of Windsor
Room 5111, Lambton Tower
hfani@uwindsor.ca
hfani.myweb.cs.uwindsor.ca

OFFICE

Tuesday-Thursday 5:30 PM-6:30 PM

WEEKLY SCHEDULE

Eastern Time	SUN	MON	TUE	WED	THU	FRI	SAT
7:00		Mark for Submissions					
		Deadline for Submissions					
16:00			Lecture A Instructor: Hossein		Lecture B Instructor: Hossein		
16:10							
16:20							
16:30							
16:40							
16:50							
17:00							
17:10							
17:20							
17:30			Office Hour A Instructor: Hossein		Office Hour B Instructor: Hossein		
17:40							
17:50							
18:00							
18:10							
18:20							
18:30							
18:40							
18:50							



Schedule_Timeline.xlsx

BB TOUR

COMP8730-1-R-2021W (Natural Lang Proc & Understanding)

Homepage
Weekly Schedule
Announcements

Learning Outcome
Course Outline (Syllabus)
Classroom

Research Project
My Grades

Discussion Board
Email
Office

Help
Virtual Classroom Tutorial

Course Management

Control Panel
Content Collection
Course Tools
Evaluation

Course Outline (Syllabus)

Attached Files
Course_Outline.pdf (296.099 KB)

Alignments

This document is the course outline (aka. syllabus). It is an essential piece that guides both the instructor and students to follow course goals, develop student learning outcomes, create and align assessment plans, as well as establish a schedule and policies for the course.

University of Windsor

School of Computer Science
Faculty of Science

COMP-8730: Natural Language Processing & Understanding
Winter 2021

Course Outline (Syllabus) v1.0	
Description	Natural language processing (NLP) is the set of methods in linguistics, computer science, and artificial intelligence concerned with how to program computers to process and analyze large amounts of natural language data. The result is a computer capable of *understanding* the contents of documents, including the language's contextual nuances within them. In the past decade, NLP has become embedded in our daily lives: Households like Alexa and Siri, automatic machine translation on the web and in social media, text classification in email inboxes, search engines, dialogue systems and