

Natural Language and Speech Processing

Lecture 1: Introduction

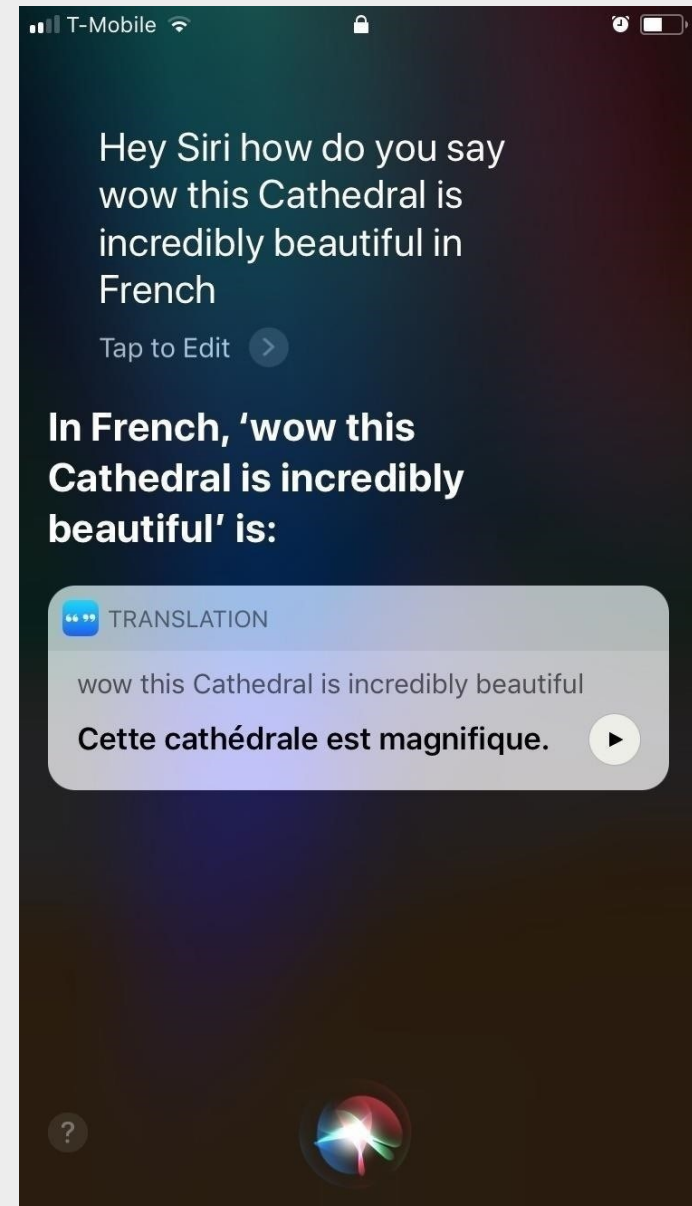
Tanel Alumäe

Goal of this lecture

- Organization of the course
 - Prerequisites
 - Course “logistics”
 - Reading material
 - Assessment
- Explain what this course is about
- What you are expected to learn

Why natural language processing for computer science students?

- Natural language based interaction with devices is becoming very wide-spread
- The most innovative companies in natural language processing research:
 - Google
 - Facebook
 - Microsoft
 - Apple
- Making natural language based computer systems is typically complicated
- But also often more profitable, as it requires more knowledge and expertise
- The course will focus on statistical/neural NLP, where models will be learned from data



Course Logistics

- Instructor: Tanel Alumäe
 - Contact: tanel.alumae@ttu.ee
 - Office phone: 620 4201
 - TTÜ PhD in 2007
 - Head of the Laboratory of Language Technology (Department of Software Science)
 - Research interests: speech recognition, other speech processing problems, deep neural networks, NLP
- One lecture and one lab exercise per week
 - Lecture: Tuesdays 14:00, ICT-A1
 - Lab: Tuesdays 16:00, ICT-122
- Register at <https://ained.ttu.ee/course/view.php?id=239>
- The course is given for the second time (be prepared for some beta-testing and open bugs)

Prerequisites

- Required:
 - Statistics, probability theory
 - Linear algebra
 - Algorithms as data structures
 - Programming (Python)
 - Tutorial: <http://cs231n.github.io/python-numpy-tutorial/>
 - Knowing some machine learning is a plus
 - Linux highly recommended for some tasks
 - Tutorial: <https://ryantutorials.net/linuxtutorial/>

Learning outcomes

- Official expected learning outcomes:
 - You are familiar with the main subfields of natural language processing (NLP) and speech processing
 - You understand the ideas behind the main NLP methods
 - You can select and apply a suitable method to solve a **new** NLP task efficiently
 - You understand the concepts behind the main speech processing tasks

Assessment

- Maximum total points: 100
- **Three homeworks**
 - Consists of both theoretical part as well as practical component (programming)
 - Each worth **15 points** of the final grade
- **Independent project**
 - Solve a practical NLP task
 - Write a report (a small research paper)
 - Worth **30 points**
- **Exam**
 - Written theoretical part
 - Interview (questions about the submitted homework and project)
 - Worth **25 points**
- In order to pass the course the student must get at least 50% from homeworks (threshold 23 points in total), at least 50% from the project (threshold 15 points) and at least 50% from the exam (threshold 13 points).

Lab exercises

- Lab exercises introduce you to some tools (mostly Python libraries) that are useful for solving practical NLP tasks
 - Pynini (for working with finite state transducers)
 - EstNLTK (for Estonian language processing)
 - spaCy (for processing other languages)
 - Sklearn (for text classification)
 - Gensim (word embeddings, topic models)
 - **Pytorch (for neural networks)**

Homeworks

- Theoretical part: e.g., analyze a certain problem, to assess your deep understanding of the topic)
- Practical part: e.g., write a simple program that handles some NLP problem (typically extends lab exercises)
- Submitted via ained.ttu.ee
- NB! Must be completed by the student alone (consulting with co-students is of course allowed)
- **Too similar submissions result in zero points for both submitters, unrecoverable**
- Plagiarism is not tolerated
 - Also applies to program code
- **NB!** I will ask questions about your submissions at the exam, to verify that you completed them yourself

Project

- The goal of the project is to give you:
 - Deeper experience in some particular NLP field
 - Experience with scientific writing
- Consists of code and report
- Report should include all the necessary components of a research paper – introduction, description of the problem, description of the solution, evaluation/experiments, conclusion
 - Analysis of previous/related work can be skipped
- You are strongly welcome to submit draft versions of reports or certain sections for feedback
- Report can be in Estonian or English, around 3000 words (template will be given)
- Submitted project will be reviewed during the exam
- To see (mostly) excellent student project samples, check <https://nlp.stanford.edu/courses/cs224n/>
- You may use external libraries (and it is encouraged), but you must acknowledge all such usages
- **Project proposal (0.5 pages) must be submitted by April 7th!**

TABLE I. DEFAULT HYPER PARAMETER VALUES.

emsize	nhid	nlayers	dropout	dropoute	dropouth	dropouti	wdrop	lr	bptt	clip	lr
300	1150	3	0.4	0.1	0.3	0.65	0.5	70	0.25	30	

TABLE II. HYPER PARAMETER SPACE.

Hyper parameter	Potential value
emsize	[300, 350, 400, 450]
nhid	[950, 1050, 1150, 1250]
nlayers	[2, 3, 4, 5]
dropout	[0.3, 0.4, 0.5, 0.6]
dropoute	[0.3, 0.4, 0.5, 0.6]
dropouth	[0.3, 0.4, 0.5, 0.6]
dropouti	[0.3, 0.4, 0.5, 0.6]
wdrop	[0.3, 0.4, 0.5, 0.6]
lr	[50, 60, 70, 80]
bptt	[0.15, 0.25, 0.35, 0.45]
clip	[20, 30, 40, 45]

surrogates proposed by [11], Gaussian Process Batch Upper Confidence Bound (GP-BUCB) [12]; an upper confidence bound-based algorithm, which models the reward function as a sample from a Gaussian process. In [13], the authors propose initializing Bayesian hyper parameters using **meta-learning**. The idea being initializing the configuration space for a novel dataset based on configurations that are known to perform well on similar, previously evaluated, datasets.

Following a meta-learning approach, we apply a genetic algorithm and a sequential search algorithm, described in the next section, initialized using the best configuration reported in [4] to search the space around optimal hyper parameters for the AWD-LSTM model. Twitter tweets collected using a geo-location filter for Nigeria and Kenya with the goal of acquiring a code-mixed text corpus serve as our evaluation datasets. We report the test perplexity distributions of the various evaluated configurations and draw inferences on the sensitivity of each hyper parameter to our unique dataset.

III. METHODOLOGY

We begin our work by establishing what the baseline and current state of the art model is for a language modeling task [4]. Applying the AWD-LSTM model, based on the open sourced code and trained on code-mixed Twitter data, we sample 84 different hyper parameter configurations for each dataset, and evaluate the resulting test perplexity distributions while varying individual hyperparameter values to understand the effect of the set of hyper parameter values selected on the model perplexity.

A. Datasets

Two sources of data are collected using the Twitter streaming API with a geolocation filter set to geo-coordinates for Kenya and Nigeria. The resulting data is code-mixed with the Kenya corpus (Dataset 1) containing several mixes of English and Swahili; both official languages in Kenya. The Nigeria data (Dataset 2) on the other hand, does not predominantly contain mixes of English with another language in the same sentence. Rather, English is simply often completely rewritten in a pidgin form. The training data for Kenya and Nigeria contains 13,185

words and 27,855 words respectively. All tweets are stripped of mentions and hashtags as well as converted to lower-case.

The phenomenon of code-mixed language use is common in locales that are surrounded by others which speak different languages or locales with a large number of immigrants. In Kenya and Nigeria as such, the use of English is influenced by the presence of one or more local languages and this is evident in the corpus.

B. Model Hyper parameters

We considered 11 hyper parameters for tuning including the size of the word embedding (emsize), the number of hidden units in each LSTM layer (nhid), the number of LSTM layers (nlayers), the initial learning rate of the optimizer (lr), the maximum norm for gradient clipping (clip), the back-propagation through time sequence length (bptt), dropout - applied to the layers (dropout), weight dropout applied to the LSTM hidden to hidden matrix (wdrop), the input embedding layer dropout (dropouth), dropout for the LSTM layer (dropouti), and dropout to remove words from embedding layer (dropoute). Table I contains the default values of the individual hyper parameters.

All experiments involved training for 100 epochs inline with available GPU resources. The training criterion was the cross-entropy loss which is the average negative log-likelihood of predicting the right next word by the LM. It took approximately two hours wall clock time to train the model for each hyper parameter configuration.

C. Sequential search

The search process begins by setting the values of each hyper parameter (configuration) to known best values (see Table I). We then iteratively search for the best value for each hyper parameter. The order used in this search is defined in the rows of Table II. Performance is evaluated based on the text perplexity for the modeling task. Once the best perplexity is identified from the list of possible values for the given hyper parameter, it is fixed and the space of the next hyper parameter in the sequence is searched. In this manner the configuration space of the model is explored.

This approach shares similarities with the method applied in [14], though it remains an open question what the impact of the sequence is on the quality of best configuration produced. For the context of this work, our aim is not to find the best configuration. Instead it is to better understand the configuration space defined by these hyper parameters to determine the impact of their values on the performance when considering a code-mixed corpora.

D. Population based search

We apply a genetic algorithm (GA) to provide a complementary approach to the sequential search for the exploration of hyper parameter configurations. The GA is a biologically

Project topics

- Predicting Tags for StackOverflow Question
 - Data: <https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction/data>
- Identifying “toxic” comments
 - Data: <https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge>
- Identify offensive social media language
 - <https://competitions.codalab.org/competitions/20011>
- Sentiment analysis for Twitter
 - SemEval-2018 Task 1: Affect in Tweets, select one of the subtasks, e.g. Task V-oc: Detecting Valence (ordinal classification)
 - <https://competitions.codalab.org/competitions/17751>
- Movie review sentiment analysis
 - <https://www.kaggle.com/c/movie-review-sentiment-analysis-kernels-only>
- Semantic similarity of texts
 - Predict which *Quora* pairs of questions contain two questions with the same meaning
 - <https://www.kaggle.com/c/quora-question-pairs>
- Community question answering
 - <http://alt.qcri.org/semeval2017/task3/>
 - Given (i) a new question and (ii) a large collection of question-answer threads created by a user community, rank the answer posts that are most useful for answering the new question
- Text normalization
 - Convert, e.g. “A baby giraffe is 6ft tall” → “A baby giraffe is **6 feet** tall”
 - English: <https://www.kaggle.com/c/text-normalization-challenge-english-language>
 - Russian: <https://www.kaggle.com/c/text-normalization-challenge-russian-language>
- Named Entity Recognition on code-switched (mixed-language) data
 - <https://competitions.codalab.org/competitions/18724>
- Identify spoken language from 10 seconds of speech
 - <https://community.topcoder.com/longcontest/?module=ViewProblemStatement&rd=16555&pm=13978>
- Propose your own (consult with me)

Project assessment

- Technical quality of writing
 - Structure
 - Description of the problem
 - Description of the solution
 - Description of the dataset
 - Presentation of results
 - Analysis and discussion
- Complexity of the used methods
 - You have to use something more than a very straightforward approach to get perfect score
- Implementation
- Results, compared to the state-of-the-art (if available)

Amount of work

- 6 ECTS = $6 \times 26 = 156$ hours of intensive work
 - On average, assuming prerequisites are mostly fulfilled
- 14 lectures = $2 \times 14 = 28$ h
- 12 lab exercises = $2 \times 14 = 28$ h
- 10 hours per homework = $3 \times 10 = 30$ h
- 60 hours for project
- 10 hours for studying for the exam
- Total: 156 hours

Dates

- Home assignments:
 - March 11
 - April 15
 - May 13
- Project:
 - May 20

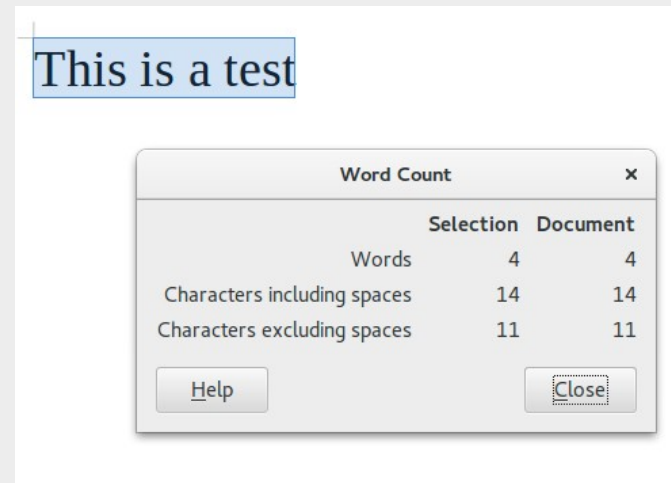
Late policy: each late day reduces points by 10%. Decay is linear. E.g., if you submit a homework 2 days late, and your real score is 11, your reduced score is $11 \times 0.80 = 8.8 \approx 9$. That is, the number of maximum late days is 10.

Reading material

- Main textbook: *Speech and Language Processing* by Jurafsky and Martin
 - Draft of 3rd edition available at <https://web.stanford.edu/~jurafsky/slp3/>
- General introduction to neural networks (free): <http://neuralnetworksanddeeplearning.com/>
- Neural networks for NLP:
 - *Neural Network Methods in Natural Language Processing* by Goldberg

What is natural language processing

- NLP is a collective term referring to automatic computational processing of human languages
- Computational techniques that use some kind of knowledge of language
 - E.g.: counting words in document (what is a *word*?)
 - “Open the pod bay doors, HAL”



(Very limited) set of sample NLP tasks

- **Information retrieval** (e.g. Google)
- **Document classification** (e.g. *sports vs business*)
- **Spelling and grammar checking and correction**
- **Machine translation**
- **Information extraction** (e.g. "*Yesterday, New York based Foo Inc. announced their acquisition of Bar Corp.*" → `Merger(Foo, Bar, date)`)
- **Natural language generation**
- **Speech recognition** (speech-to-text)
- **Speech synthesis** (text-to-speech)
- **Speaker recognition** (who speaks?)

Space Odyssey 2001

- Let's look at the *Space Odyssey 2001* example in more detail:
<https://youtu.be/dSIKBlibolo?t=52>



*Open the pod
bay doors, HAL*

NLP components of HAL

*I'm sorry Dave,
I'm afraid I can't
do that*

- Speech recognition (speech-to-text)
- Audio-visual speech recognition (lip reading)
- Speech synthesis (text-to-speech)
- Analyze the structure of human language
 - Dialog act type identification (identify that the utterance is a request vs a simple statement)
 - Syntax analysis (identify that the “pod bay doors” must be “opened”)
 - Semantic analysis (map “pod bay doors” and “open” to certain objects and actions that HAL knows about)
- Natural language generation
 - Discourse (use of “that” in the response, vs “bay doors”)
 - Pragmatics (use of “I’m afraid I can’t do that” vs “No”)

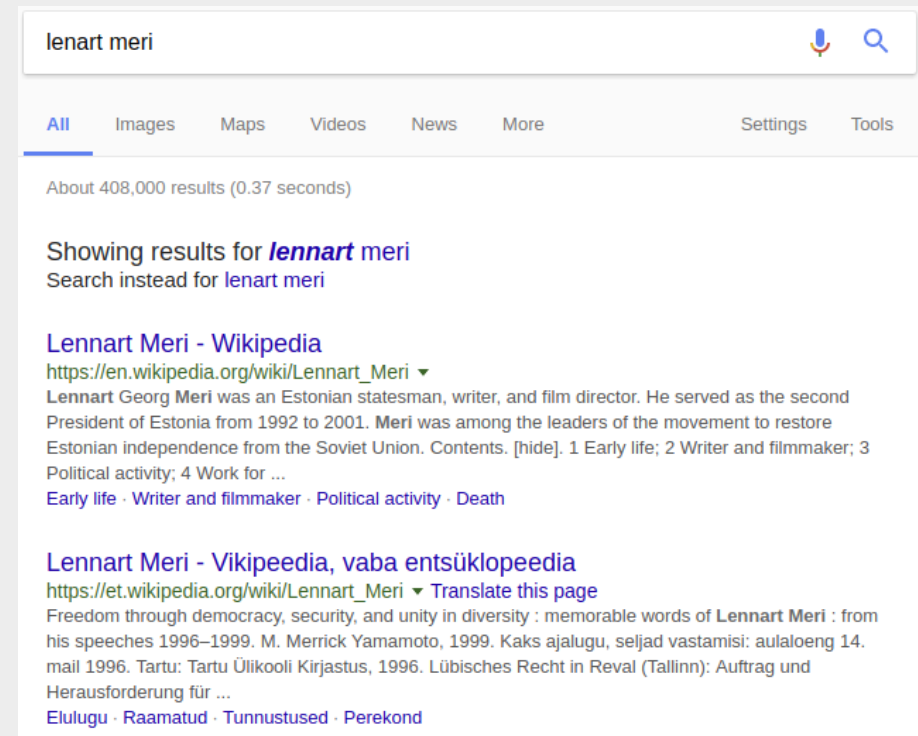
Jeopardy!

- <https://youtu.be/P0Obm0DBvwl?t=227>
- IBM Watson's AI won the best US human players in 2011
- NLP components:
 - Syntactic and semantic analysis (to understand the question)
 - Information retrieval and extraction
 - Natural language generation
 - Speech synthesis



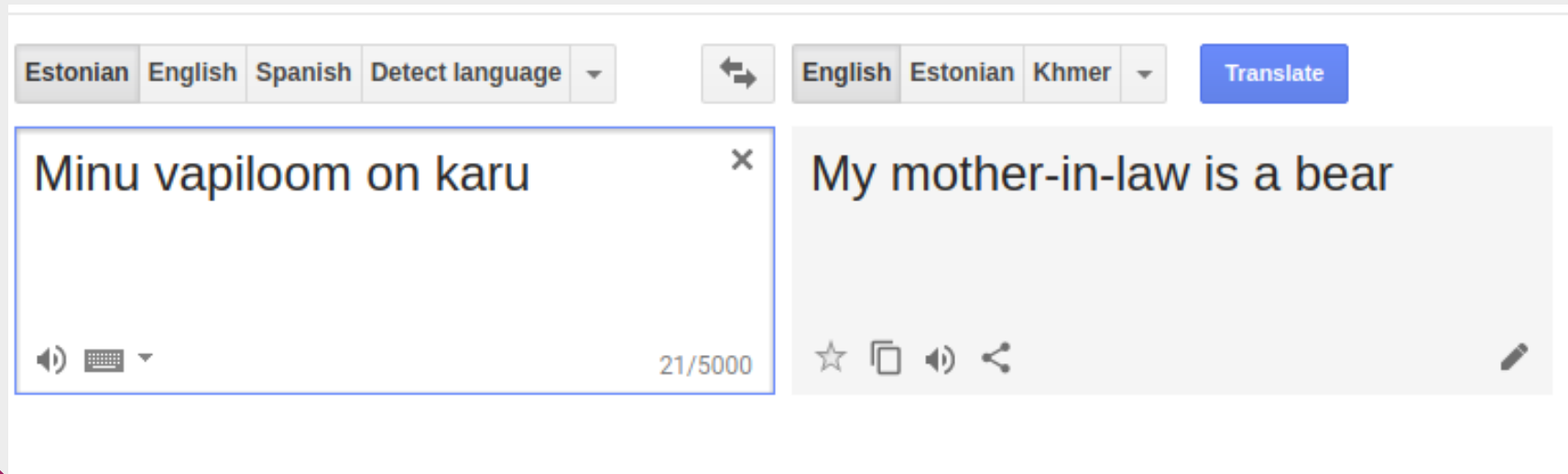
Information Retrieval

- Retrieving and ranking documents, given a search query (e.g. Google)
- Possible NLP components:
 - Spell-check and auto-correct the query
 - Lemmatize the query and documents (e.g., for a query *pöial*, also show documents containing *pöidla*)
 - Synonyms and closely related terms, e.g. for query *mad cow disease* also show documents containing *BSE* and *bovine spongiform encephalopathy*



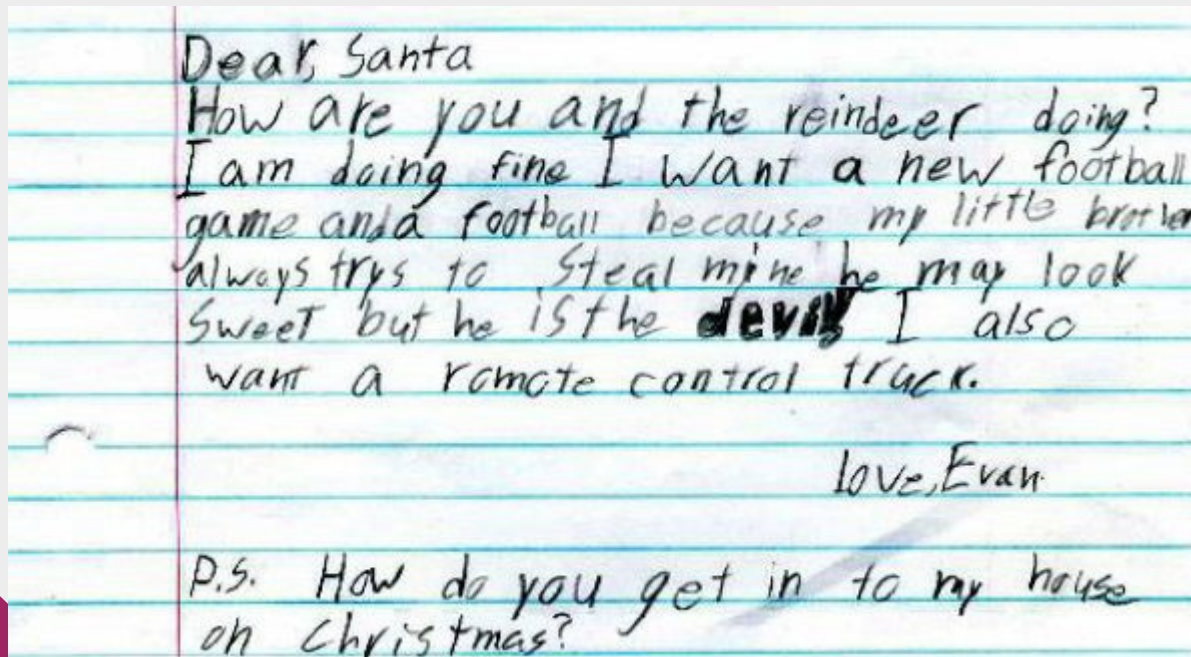
Machine translation

- Used as standalone, or as an aid for human translators
- Great progress in recent years :)



Information extraction

- Finding structured information from unstructured or semi-structured text



Dear Santa
How are you and the reindeer doing?
I am doing fine I want a new football
game and a football because my little brother
always trys to steal mine he may look
sweet but he is the **devil** I also
want a remote control truck.

Love, Evan

P.S. How do you get in to my house
on Christmas?

→ Football game
Football
Remote control truck

Search from audio documents

- Finding relevant content from audio archives is awkward and time-consuming
- Usually relies on meta-data (titles, short summary, etc)
- Automatically generated speech transcripts make searching possible
- Compare:
 - <https://arhiiv.err.ee/err-audioarhiiv>
 - <http://bark.phon.ioc.ee/tsab>

Why is processing natural language difficult?

- Full human-level error-free NL is **AI-complete**: it requires human-level AI
- It is difficult to represent all the different meanings in natural language abstractly:
 - *Tallinn is the capital of Estonia* → Capital(Estonia, Tallinn)
 - *Jerusalem is key to any peace agreement* → ???
- Lots of relationships between concepts
- Different ways to represent the same meaning:
 - *Firefighter dies battling huge California wildfire*
 - *Firefighter dies, thousands more take on California blaze*
 - *Cal fire family mourns San Diego firefighter killed in Thomas fire*
 - *Firefighter killed battling Thomas fire*



Why is processing natural language easy?

- Language is highly redundant:
 - E.g. movie review: *I hated this movie. Hated hated hated hated this movie. Hated it. Hated every simpering stupid vacant audience-insulting moment of it. Hated the sensibility that thought anyone would like it. Hated the implied insult to the audience by its belief that anyone would be entertained by it.* → **Rating: 1/5**
- Many crude methods provide surprisingly good results

Ambiguity

Natural language is often highly ambiguous at many levels

- Speech:

- *The tail of the dog ↔ The tale of the dog*
- *It's easy to recognize speech ↔ It's easy to wreck a nice beach ↔ It's easy to wreck an ice beach*
- Estonian: *sõidutasime autoga ↔ sõidutas imeautoga*

- Lexical: e.g. *back*: noun (*my back*) vs adverb (*to back away*) vs adjective (*back door*)

- Syntactic: *I heard his cell phone ring **in my office***

- Semantic: *I don't like it when my father **smokes***

- Metonymy: **White House** said to be considering replacing Tillerson with CIA chief

- Metaphors: *food for thought, lose one's way*

Models and algorithms

- Computer science
 - Finite-state automata and transducers
 - (Hidden) Markov models
 - Regular grammars, context-free grammars
 - Search algorithms (e.g. depth-first search, A* search)
- Probability theory
- Machine learning
 - Naive Bayes models
 - Maximum entropy models
 - Neural networks
- Linguistics
- Phonetics
- Signal processing

History of NLP

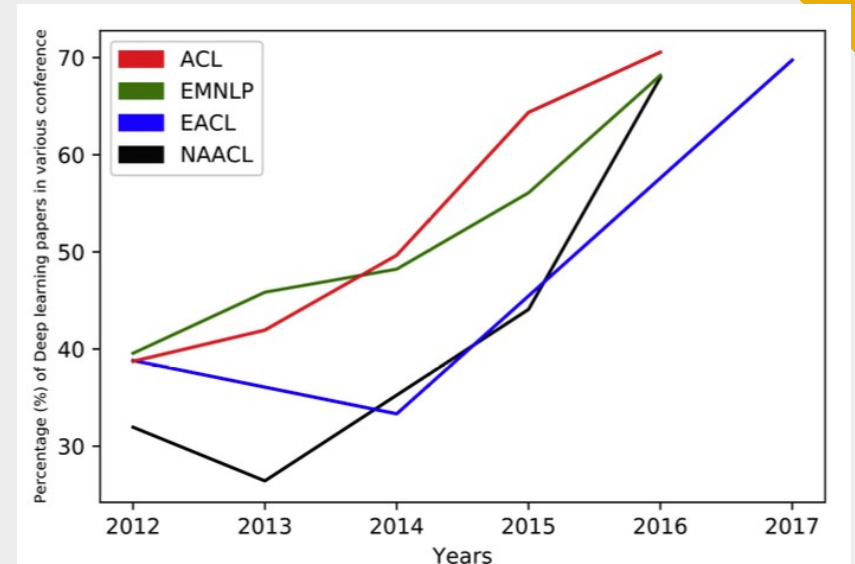
- 1940-1950s: foundational insights
 - Automata
 - Noisy channel model
 - Spectrograph
- 1957-1970: symbolic vs probabilistic approach
- 1970-1983: explosion in research, development of several paradigms that still dominate

History of NLP, II

- 1983-1993:
 - Finite state methods further developed
 - Rise of probabilistic models
 - Frederick Jelinek, head of IBM's speech recognition group, 1985: *"Every time we fire a phonetician/linguist, the performance of our system goes up"*
- 1994-2010
 - Probabilistic methods everywhere
 - Faster computers result in many successful applications (speech recognition, grammar checking)
 - Internet – vast amounts of unstructured text, rise of social networks
 - Bill Gates, 1997: *"In this 10-year time frame, I believe that we'll not only be using the keyboard and the mouse to interact, but during that time we will have perfected speech recognition and speech output well enough that those will become a standard part of the interface."*

History of NLP, III

- 2010-
 - **Deep neural networks** everywhere
 - Smartphones
 - Siri, Google Assistant, Cortana
 - **Amazon Echo** fulfills Bill Gates' dream

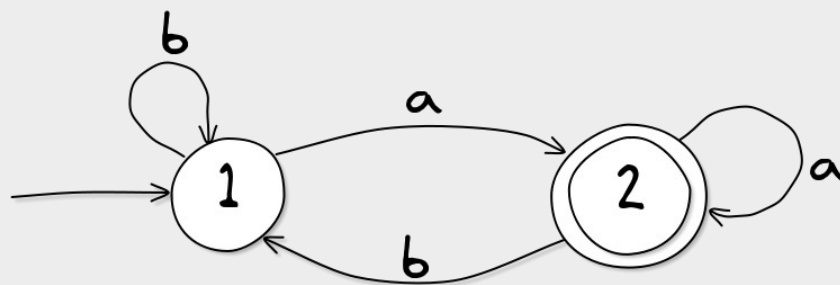


Course contents

- Finite state automata, regular expressions
- Morphology
- N-gram language models
- Text classification with statistical models
- Word classification
- Word embeddings, neural network language models
- Convolutional neural network models for sentences
- Recurrent neural networks for NLP
- Speech recognition
- Speaker and language recognition
- Machine translation

Finite state automata, regular expressions

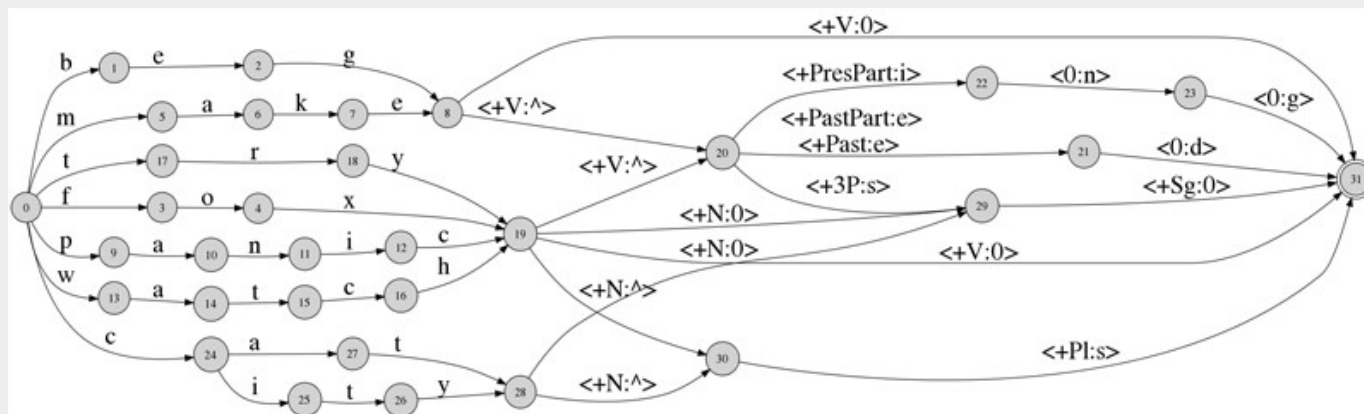
Symbol	Example	Definition
.	. ₀₀ , ₀₀ A. ₀₀	match any character
[]	[ABCabc] ₀₀ , ₀₀ [A-Ca-c]	match any char in the list
[^]	[^Z] ₀₀ , ₀₀ [^XYZ] ₀₀ , ₀₀ [^x-z]	match all except the chars in the list
~	~C ₀₀ , ₀₀ ~A.*	next token must be the first part of string
\$	[CO]G\$	prev token must be the last part of string
*	C* ₀₀ , ₀₀ [ab]*	match 0 or more copies of prev char or regular expression token
+	C* ₀₀ , ₀₀ [ab]+	match 1 or more copies of the prev token
	C O	match either the 1st token or the 2nd
\(\)	\(CA\)*	combines multiple tokens into one



Morphology, finite state transducers

- Morphological parsing and generation

Input word	Output morphemes
cats	cat +N +PL
cat	cat + N + SG
cities	city + N + PL
walks	walk + V + 3SG
cook	cook +N +SG or cook +V



N-gram language models

- Predict the next word, given $N-1$ previous words

<s>	must	kass	jooksis	üle	tee	</s>
	ja	valge	jooksis	üle	selja	</s>
	ta	oli	</s>	</s>	tee	ja
	ma	sõstra	ja	ta	välja	seda
	aga	on	oli	mööda	keha	oli
	see	miljon	on	ja	</s>	ääres
	kui	mees	kes	läbi	jäänud	peal
	kas	kass	üumber	trepist	pea	rada

Text classification

- Is tweet's emotion positive or negative (or neutral)?
- Spam filtering
- Detection of violent content, abuse
- E-mail sorting

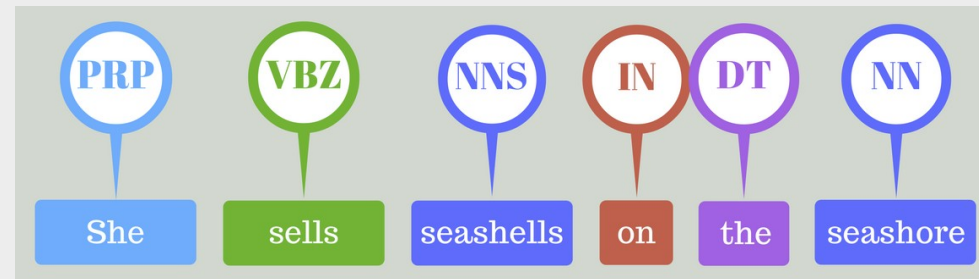
Word classification with statistical models

- Named entity recognition
- Morphological tagging

In 1917, Einstein applied the general theory of relativity to model the large-scale structure of the universe. He was visiting the United States when Adolf Hitler came to power in 1933 and did not go back to Germany, where he had been a professor at the Berlin Academy of Sciences. He settled in the U.S., becoming an American citizen in 1940. On the eve of World War II, he endorsed a letter to President Franklin D. Roosevelt alerting him to the potential development of "extremely powerful bombs of a new type" and recommending that the U.S. begin similar research. This eventually led to what would become the Manhattan Project. Einstein supported defending the Allied forces, but largely denounced using the new discovery of nuclear fission as a weapon. Later, with the British philosopher Bertrand Russell, Einstein signed the Russell-Einstein Manifesto, which highlighted the danger of nuclear weapons. Einstein was affiliated with the Institute for Advanced Study in Princeton, New Jersey, until his death in 1955.

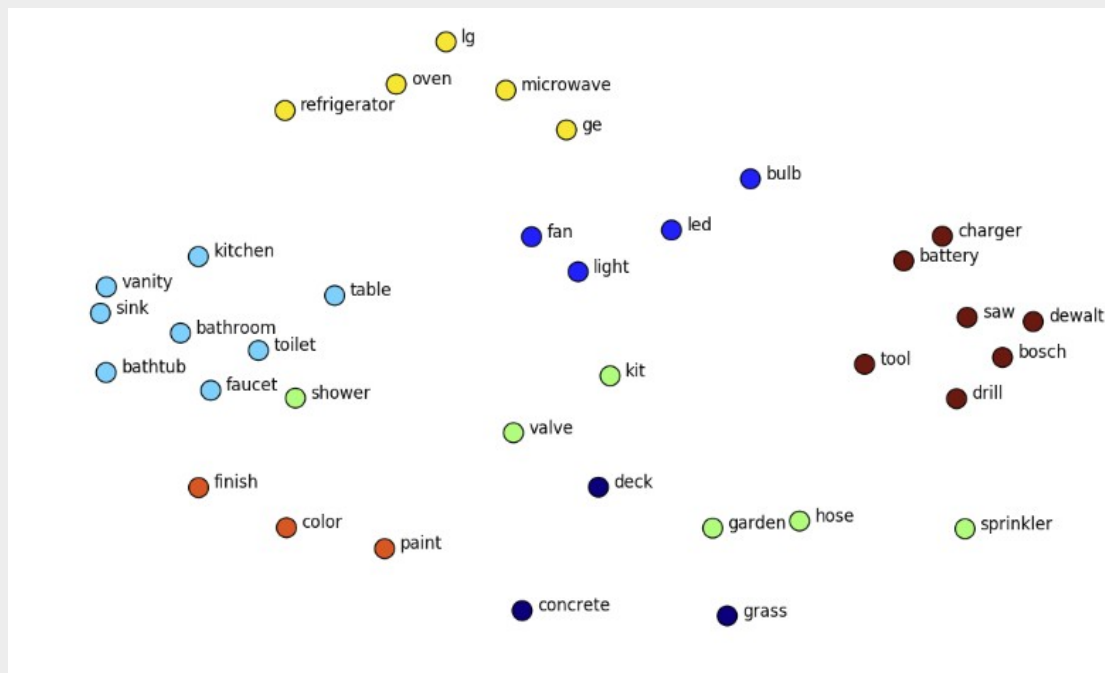
Tag colours:

LOCATION TIME PERSON ORGANIZATION MONEY PERCENT DATE

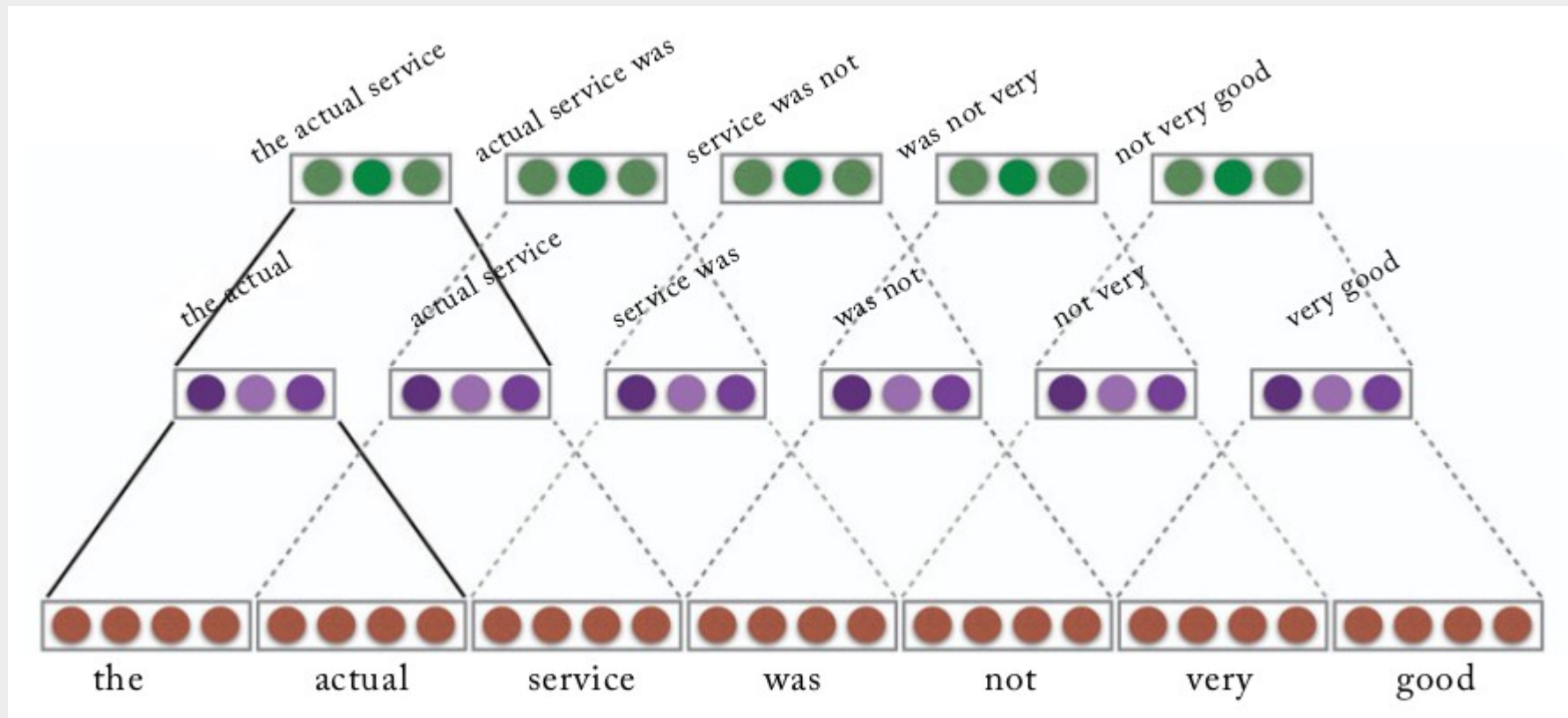


Word embeddings

- Automatically projecting *words* into low-dimensional space, using only the statistics of their contexts that they appear in a large text corpus
- Semantically and/or syntactically similar words are mapped to nearby points

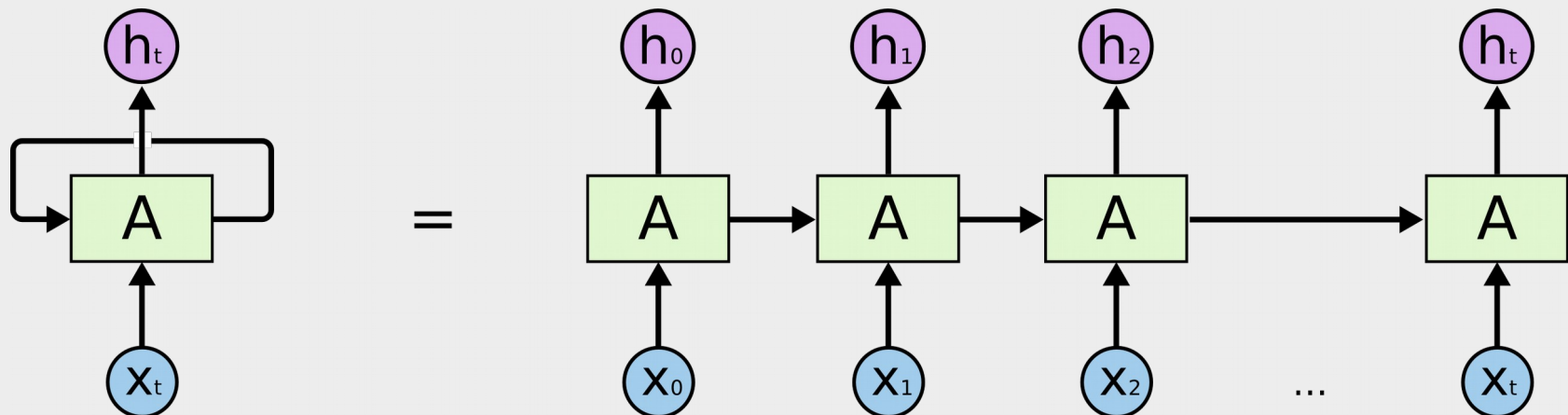


Convolutional neural networks for NLP



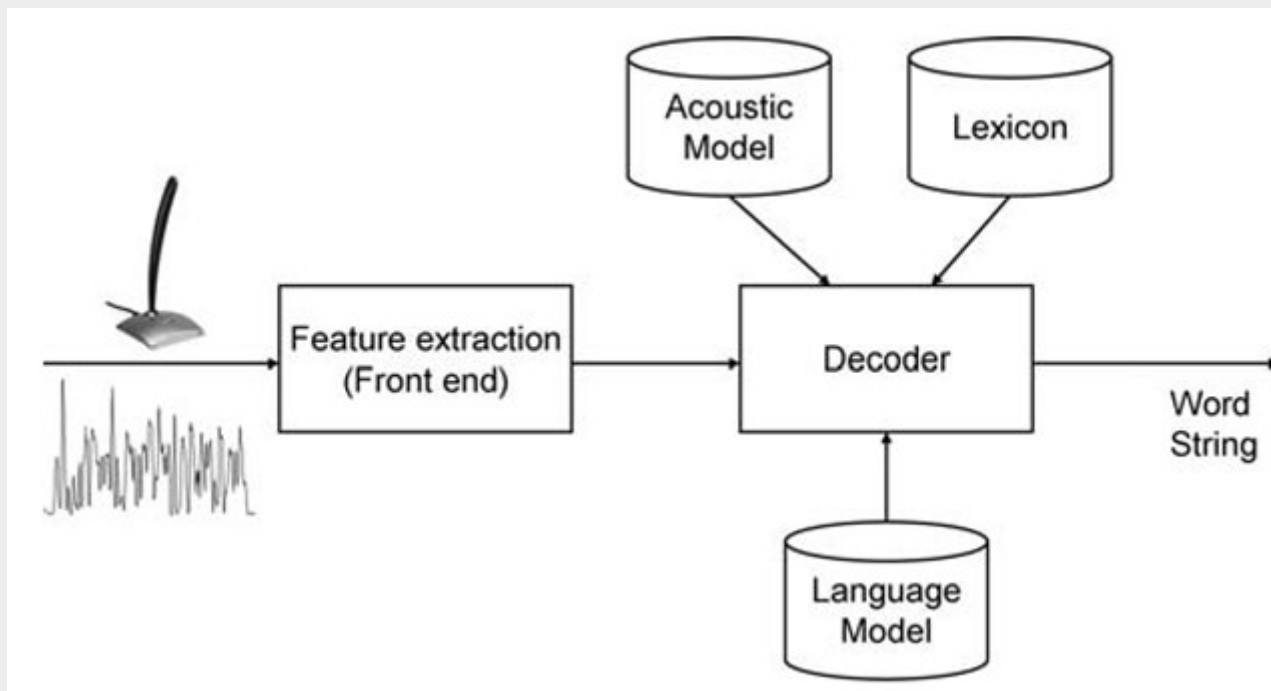
Recurrent neural networks for NLP

- Probably the most widely used model type in today's NLP research
- Can be used for:
 - Text classification
 - Word classification
 - Machine translation
 - Summarization
 - ...



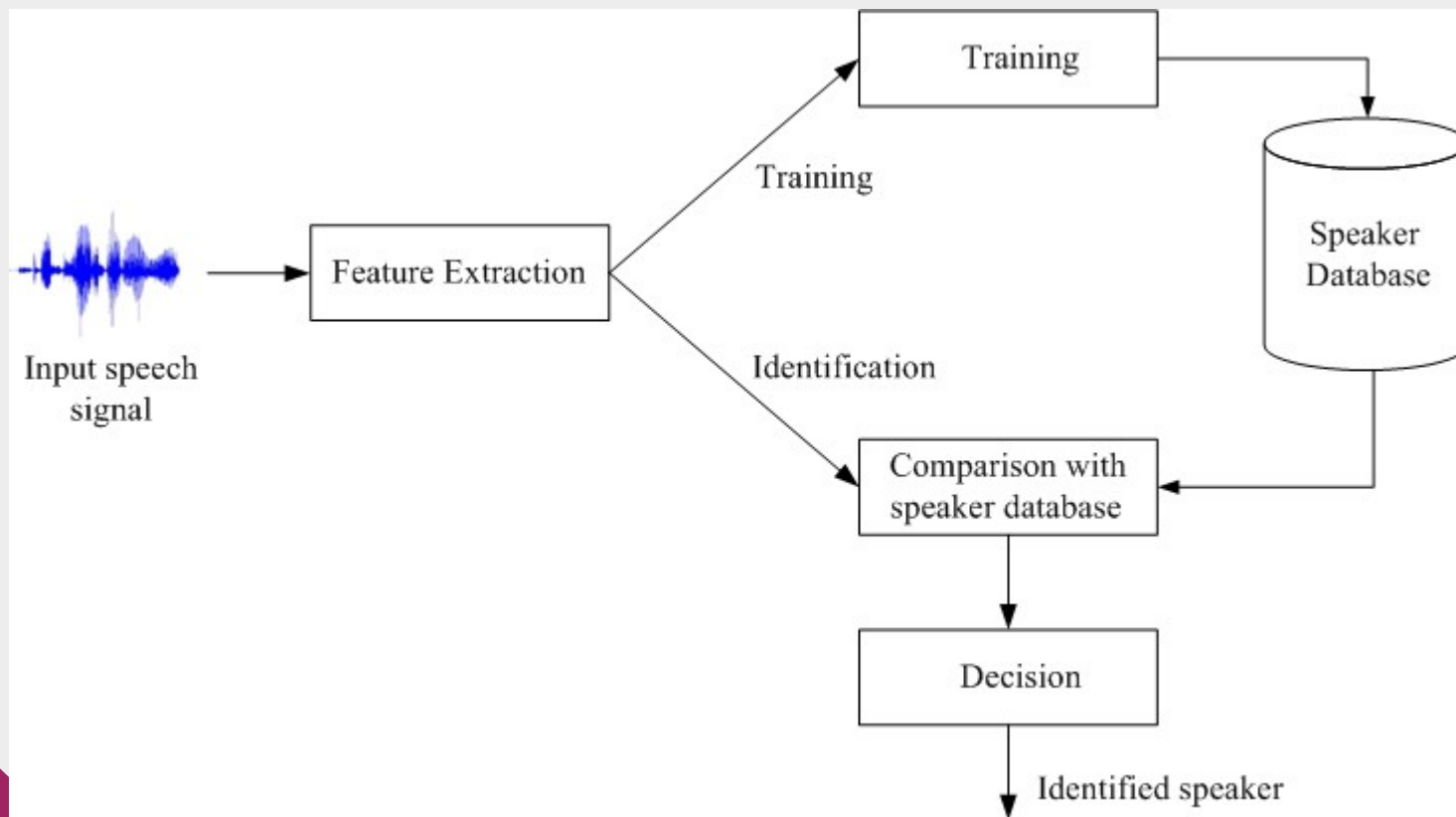
Speech recognition

- Converting speech into text



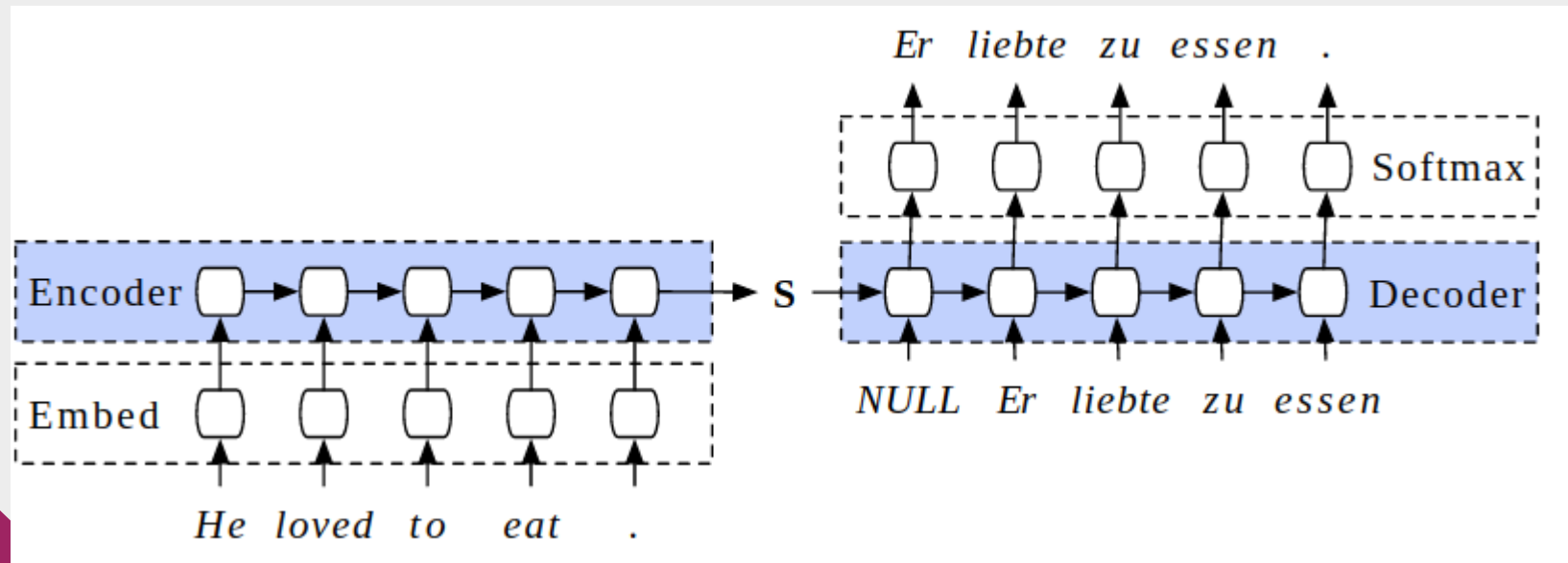
Speaker and language identification

- Identifying speaker or spoken language (e.g., Estonian, English or Russian?) from speech)



Machine Translation

- Neural machine translation
- One recurrent neural network is used to compress the source language sentence and another RNN to generate the target language sentence from the compressed representation
- Many tricks needed to get this to work



Summary

- Natural language processing is difficult
- But very interesting
- Nowadays, many new methods and ideas in AI (particularly deep learning) come from the NLP community
- Learning modern NLP gives you a good background for doing any machine learning or data science work professionally
- Interesting Master thesis topics available (contact me as soon as possible if interested – I have limited supervising capacity)