An introduction to NLP and Text Mining

By: **Farrokh Mehryary**, TurkuNLP group, University of Turku
Email: farmeh@utu.fi

A guest lecture for "Artificial Intelligence and Big Data" module
in the Programme of Computer Applications at Häme University of Applied Sciences.

October 2021

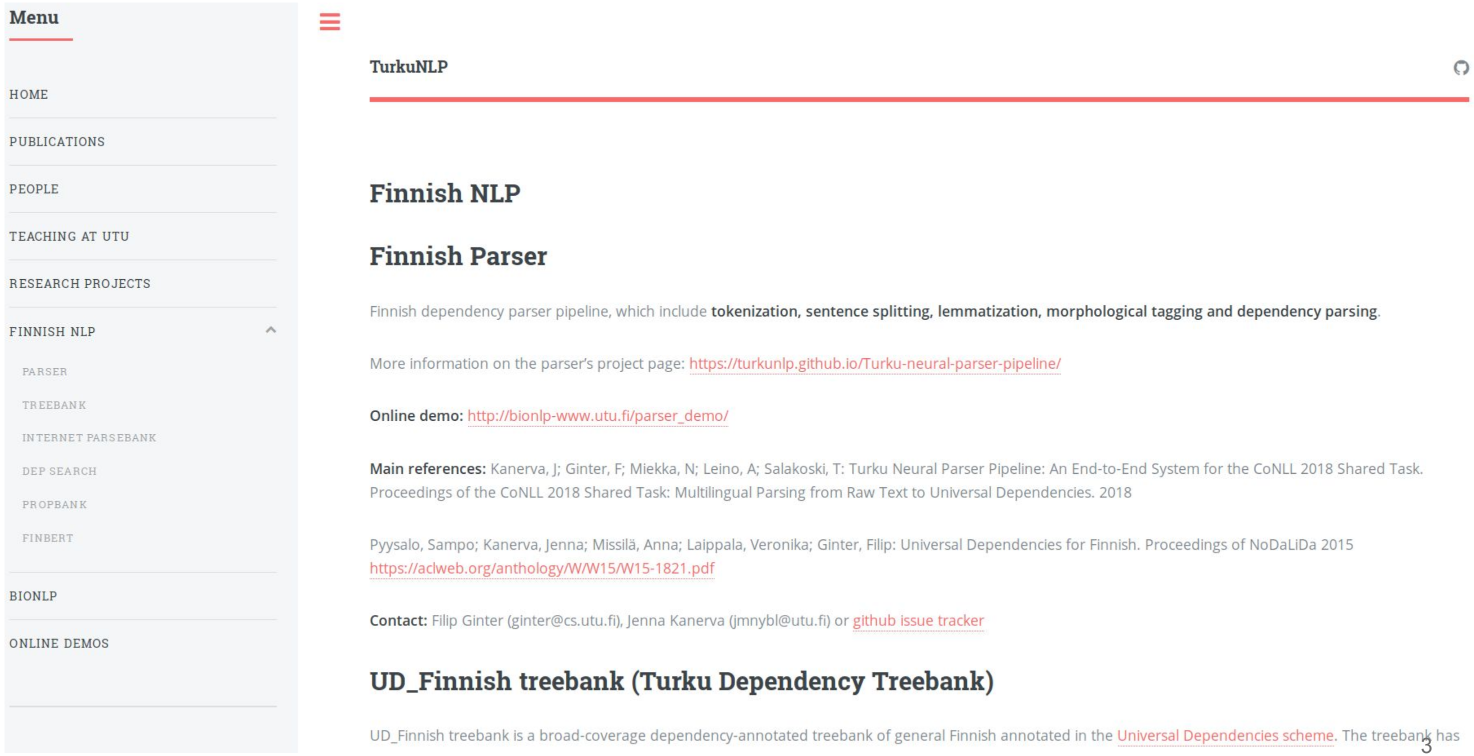
Who am i?

- Last year Phd student in TurkuNLP group, University of Turku,
- AI scientist working for Silo.ai and NLP advisor for Riskthinking.ai

Focus: Biomedical natural language processing (BioNLP) and Text Mining using deep learning-based methods

- I have achieved numerous high ranks in international BioNLP competitions (BioCreative, BioNLP, CAFA, etc).
- TurkuNLP:
  - 10+ researchers
  - Wide scope (Finnish texts, English texts, Biomedical Texts, Clinical texts)
  - 250+ publications in the field
  - Numerous international 1st ranks in various competitions

turkunlp.org



Finnish NLP

Finnish Parser

Finnish dependency parser pipeline, which include **tokenization, sentence splitting, lemmatization, morphological tagging and dependency parsing**.

More information on the parser's project page: https://turkunlp.github.io/Turku-neural-parser-pipeline/

Online demo: http://bionlp-www.utu.fi/parser_demo/

Main references: Kanerva, J; Ginter, F; Miekka, N; Leino, A; Salakoski, T: Turku Neural Parser Pipeline: An End-to-End System for the CoNLL 2018 Shared Task. Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. 2018

Pyysalo, Sampo; Kanerva, Jenna; Missilä, Anna; Laippala, Veronika; Ginter, Filip: Universal Dependencies for Finnish. Proceedings of NoDaLiDa 2015 https://aclweb.org/anthology/W/W15/W15-1821.pdf

Contact: Filip Ginter (ginter@cs.utu.fi), Jenna Kanerva (jmnybl@utu.fi) or github issue tracker

UD_Finnish treebank (Turku Dependency Treebank)

UD_Finnish treebank is a broad-coverage dependency-annotated treebank of general Finnish annotated in the Universal Dependencies scheme. The treebank has

UNIVERSITY OF TURKU

Content

- NLP, computational linguistics, text mining
- NLP v.s general machine learning
- NLP systems
- NLP tasks

NLP, Computational Linguistics, Text mining

- **NLP:** processing textual data for different useful applications, i.e., text summarization, automatic machine translation, information retrieval, information extraction, sentiment analysis, document clustering, topic modeling, etc.

  Useful for diffedifferent applications in **science and academia, industry, politics, economy, business, humanities, etc.**

- **Computational linguistics:** uses the same methods as NLP **with the aim of studying the language**, i.e., how a language has evolved over time, how social media language differs from formal texts, what are the differences between male and female teenagers' posts on social media, etc.

- **Text mining:** a branch of NLP which aims to extract useful information from large text corpora.

NLP vs general machine learning

General machine learning uses structured (table data):
Example: A table of (previous/current) customers' information

	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	Y
C1															Yes
C2															No
C3															No
C4															Yes
C5															No

Example: A table of (previous/current) customers' information

Features (Columns f1, ..., fn): gender, age, education, gross income, net income, #children, existing loan total, existing debt payment monthly, nationality, marital status, etc...

Supervised learning: Using the information of customers from past, predict the future.

- if a new customer is low_risk/high_risk (**binary** classification)
- If a new customer is low_risk/medium_risk/high_risk (**multi-class** classification)
- types of products a customer may be interested to buy next (**multi-label** classification)
- Maximum amount loan in Euros to offer which is low risk (**regression:** positive integer, x>=0)

Unsupervised learning: No labels (column Y) exists yet. Try to extract other useful information:

- Partition the customers into different groups (clusters) so that similar customers end-up in similar groups (**Clustering)**.
    - We can give different weights to different features.
    - We can use different clustering algorithms and similarity measures/

NLP vs general machine learning

- **General machine learning conventionally relies on structured data.**

- **Unstructured data:**
  - **Graph data** ➔ graph processing (e.g., graph mining, predicting missing edges/weights, etc)
  - **Digital images** ➔ image processing and mining
  - **Digital sound** (music, speech, ...) ➔ speech recognition, finding similar music, etc
  - **Digital videos** ➔ video processing
  - **Signals** ➔ signal processing
  - **Natural language** (**human language**) ➔ **N**atural **L**anguage **P**rocessing: processing textual data for different aims: text mining, automatic translation, information retrieval, text summarization, etc
  - **Computer programming languages:** suggest corrections, suggest similar codes, suggest improvements, etc.
  - etc

NLP vs general machine learning

- **Processing unstructured data is tricky!**

is_dog
not_dog
?
?
?

Image from: AI & CYBERSECURITY MOOC (https://digicampus.fi/course/view.php?id=1391)

- In tables, the smallest unit of data is a feature (numerical or categorical, e.g., income or gender) ➔ **informative and useful**!
- In images, the smallest unit of data is a pixel (RGB) ➔ **not very useful! Why?**

NLP vs general machine learning

- **Processing unstructured data is tricky!**

Sally Anne Thompson, Animal Photography | Ron Wilbie, Animal Photography

Answer:

Even two very similar dog images (to human eyes), vary a lot in their raw format! (matrix of pixels)!

- Not suitable for supervised learning!
- Not suitable for unsupervised learning!

NLP vs general machine learning

What to do with unstructured data?

1) **Feature-based machine learning methods:**
   Relies on manually engineered features to transform the raw data into a set of features, suitable for the machine learning task at hand.

   Raw data ➔ **Feature extraction** ➔ a set features (i.e., a new **representation** of the original data) ➔ use this in traditional machine learning algorithms (SVMs, K-means clustering, etc).

   ➔ Based on : **Feature engineering** and **feature extraction**!

2) **Modern deep learning-based methods:**
   Directly uses the data in its **raw format**, but relies on deep neural networks architectures which are able to **automatically learn efficient representations** of data items.

   ➔ Based on: **Automatic representation learning**!

NLP vs general machine learning

An example: train a computer to recognize if an image is a dog or not? (binary classification).

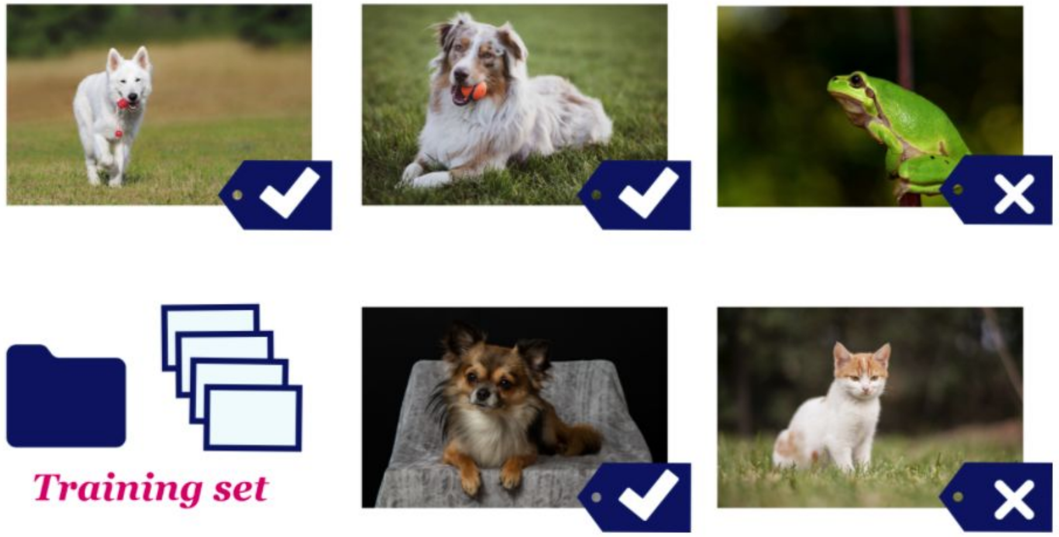


Image from: AI & CYBERSECURITY MOOC (https://digicampus.fi/course/view.php?id=1391)

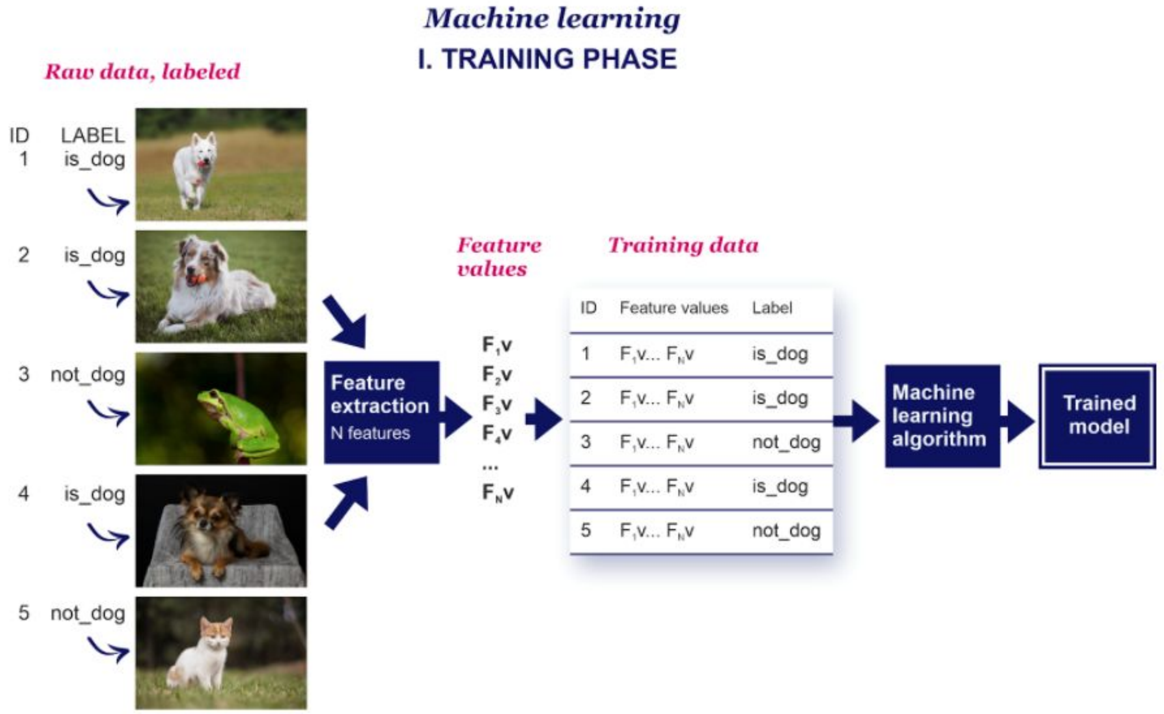
Start by building a representative training set, hand labeling each image (manual annotation).

All standard machine learning techniques applies:

- Split data into train, devel, test
- Cross-validation folds
- Make sure you have enough data
- Etc

NLP vs general machine learning

An example: train a computer to recognize if an image is a dog or not? (binary classification).



ID	Feature values	Label
1	F_1v... F_Nv	is_dog
2	F_1v... F_Nv	is_dog
3	F_1v... F_Nv	not_dog
4	F_1v... F_Nv	is_dog
5	F_1v... F_Nv	not_dog

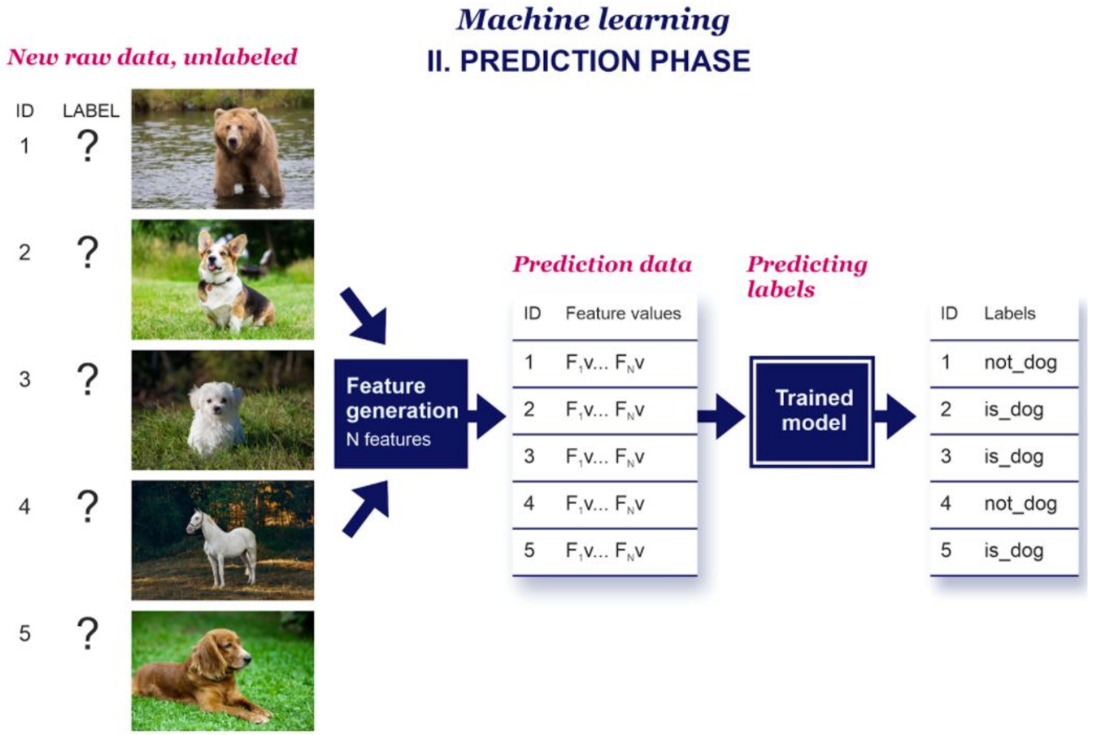
Features:

- Different types of corners
- Different types of contours
- Different types of shapes
- Different types of patterns

Image from: AI & CYBERSECURITY MOOC (https://digicampus.fi/course/view.php?id=1391)

NLP vs general machine learning

An example: train a computer to recognize if an image is a dog or not? (binary classification).



ID	LABEL
1	?
2	?
3	?
4	?
5	?

ID	Feature values
1	$F_1v \dots F_Nv$
2	$F_1v \dots F_Nv$
3	$F_1v \dots F_Nv$
4	$F_1v \dots F_Nv$
5	$F_1v \dots F_Nv$

ID	Labels
1	not_dog
2	is_dog
3	is_dog
4	not_dog
5	is_dog

Image from: AI & CYBERSECURITY MOOC (https://digicampus.fi/course/view.php?id=1391)

NLP vs general machine learning

An example: train a computer to recognize if an image is a dog or not? (binary classification).

- **Feature engineering** plays a crucial role in feature-based machine learning.
- The performance of machine learning learning system (e.g., f1-score), heavily depends on how good the feature generation component works (what features are extracted and how suitable they are).

➜ **Feature-based machine learning methods heavily rely on feature engineering!**

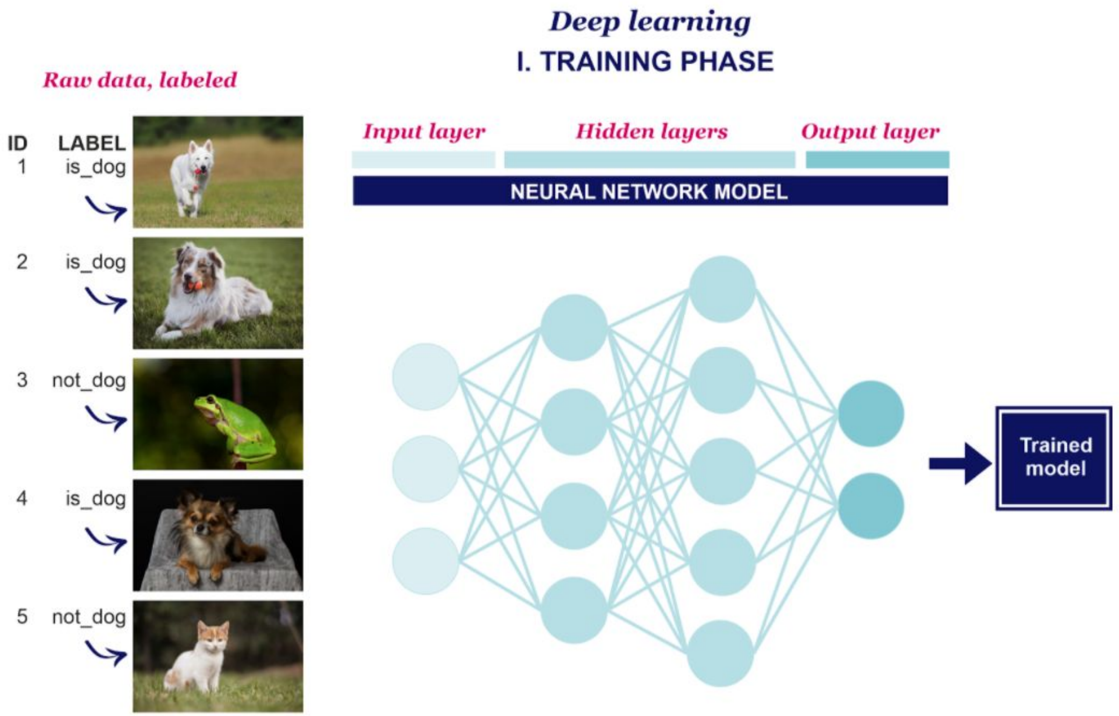
Problems:

1) design and implementation of features is a very time consuming process!
2) It requires different types of expertise (domain expertise, mathematics, programming).
3) Features for one problem/data-type may not be suitable for other problems/data-types.

Farrokh Mehryary (farmeh@utu.fi)

NLP vs general machine learning

An example: train a computer to recognize if an image is a dog or not? (binary classification).

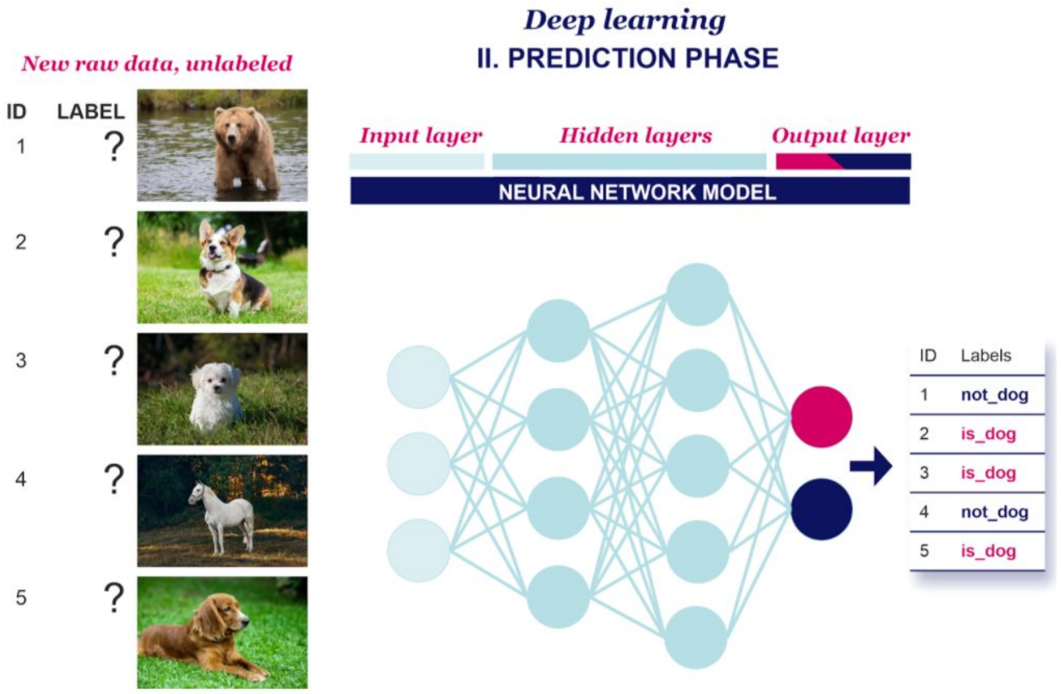


- Raw data → Input layer
- Output (decision) layer → Prediction(s)

Image from: AI & CYBERSECURITY MOOC (https://digicampus.fi/course/view.php?id=1391)

NLP vs general machine learning

An example: train a computer to recognize if an image is a dog or not? (binary classification).



ID	LABEL
1	?
2	?
3	?
4	?
5	?

ID	Labels
1	not_dog
2	is_dog
3	is_dog
4	not_dog
5	is_dog

- Raw data → Input layer
- Output (decision) layer → Prediction(s)

Image from: AI & CYBERSECURITY MOOC (https://digicampus.fi/course/view.php?id=1391)

NLP vs general machine learning

Deep learning allows computers to **automatically** learn efficient representations from the raw data, that is suitable for the particular machine learning task at hand.

In deep learning, computer learns more complex concepts out of simpler ones, in subsequent layers in the neural network.

For example:

- In image recognition, Faces are composed shapes, shapes composed of edges, etc.

- In NLP, a page is composed of paragraphs, a paragraph is composed of sentences, a sentences is composed of phrases, phrases are composed of words, and words are composed of characters.

NLP vs general machine learning

Deep learning allows computers to **automatically** learn efficient representations from the raw data, that is suitable for the particular machine learning task at hand.

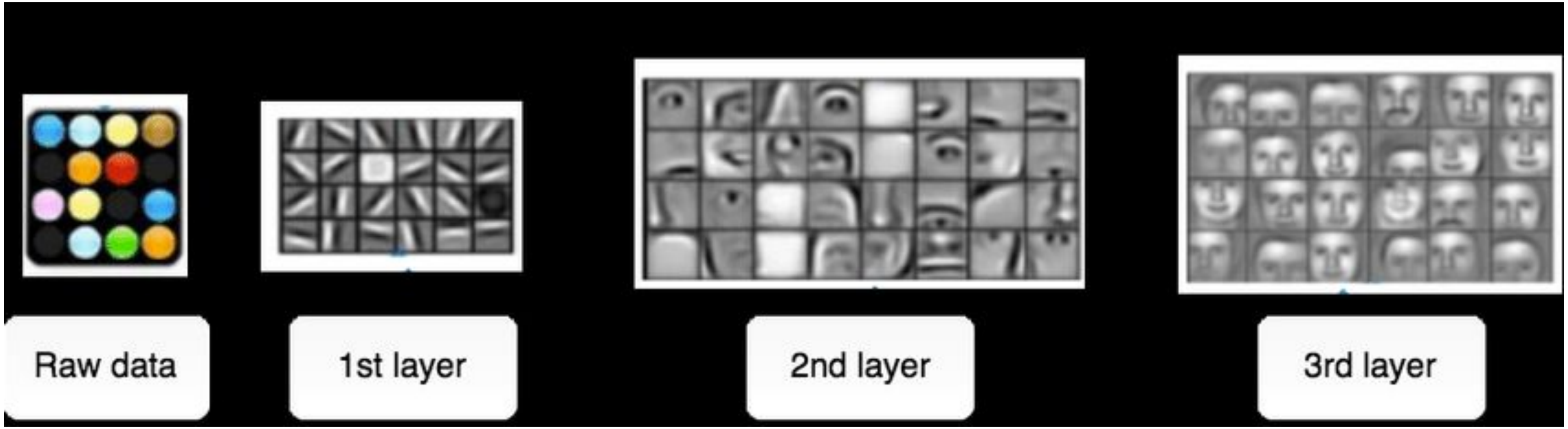


Image from Research gate

NLP vs general machine learning

Conclusion:

- In feature-based methods, we use feature-engineering to **transform** the data from its **raw format** into a **representation** (a set of features that are extracted from the raw data), that is suitable for the machine learning task at hand.

  Better features → Better representations → Better ML performance

- In deep learning-based methods, neural network automatically learns an adequate **representation** of the raw data, that is suitable for the machine learning task at hand. Although a neural network architecture may have many hidden layers, the last hidden layer (before the output layer) produces the final representation of the raw input.

- Better neural network architectures → Better representations → Better ML performance

NLP vs general machine learning

NLP deals with texts written in natural languages (e.g., English, Finnish, Swedish, etc).

- Text is unstructured data, so we need to either use **feature-based** or **deep learning-based** machine learning methods for text.

- Based on application and usage, we can look at a written text as a
    - **sequence of words**
    - **sequence of characters**

- **Remember**: elements in a sequence are ordered, whereas elements in a set do not have any order.

NLP vs general machine learning

- Based on application, we can look at a written text as a **sequence of words**, or **characters**, or **both**.

```
>>> import spacy
>>> mytext = "The University of Turku is an active academic community of 25000 students and personnel.\nWe study, teach, and
work for a better future."
>>> nlp_pipeline = spacy.load("en_core_web_sm")
>>> doc = nlp_pipeline(mytext)
>>> print ([x.text for x in doc])

['The', 'University', 'of', 'Turku', 'is', 'an', 'active', 'academic', 'community', 'of', '25000', 'students', 'and',
'personnel', '.', '\n', 'We', 'study', ',', 'teach', ',', 'and', 'work', 'for', 'a', 'better', 'future', '.']

>>> print ([x for x in mytext])

['T', 'h', 'e', ' ', 'U', 'n', 'i', 'v', 'e', 'r', 's', 'i', 't', 'y', ' ', 'o', 'f', ' ', 'T', 'u', 'r', 'k', 'u', ' ',
'i', 's', ' ', 'a', 'n', ' ', 'a', 'c', 't', 'i', 'v', 'e', ' ', 'a', 'c', 'a', 'd', 'e', 'm', 'i', 'c', ' ', 'c', 'o', 'm',
'm', 'u', 'n', 'i', 't', 'y', ' ', 'o', 'f', ' ', '2', '5', '0', '0', '0', ' ', 's', 't', 'u', 'd', 'e', 'n', 't', 's', ' ',
'a', 'n', 'd', ' ', 'p', 'e', 'r', 's', 'o', 'n', 'n', 'e', 'l', '.', '\n', 'W', 'e', ' ', 's', 't', 'u', 'd', 'y', ',', '
', 't', 'e', 'a', 'c', 'h', ',', ' ', 'a', 'n', 'd', ' ', 'w', 'o', 'r', 'k', ' ', 'f', 'o', 'r', ' ', 'a', ' ', 'b', 'e',
't', 't', 'e', 'r', ' ', 'f', 'u', 't', 'u', 'r', 'e', '.']
```

NLP vs general machine learning

Some features of textual data:

- In structured data (tables), data elements can be **continuous** (like height, weight, temperature), but in written texts, input items (words or characters) are **discrete**.

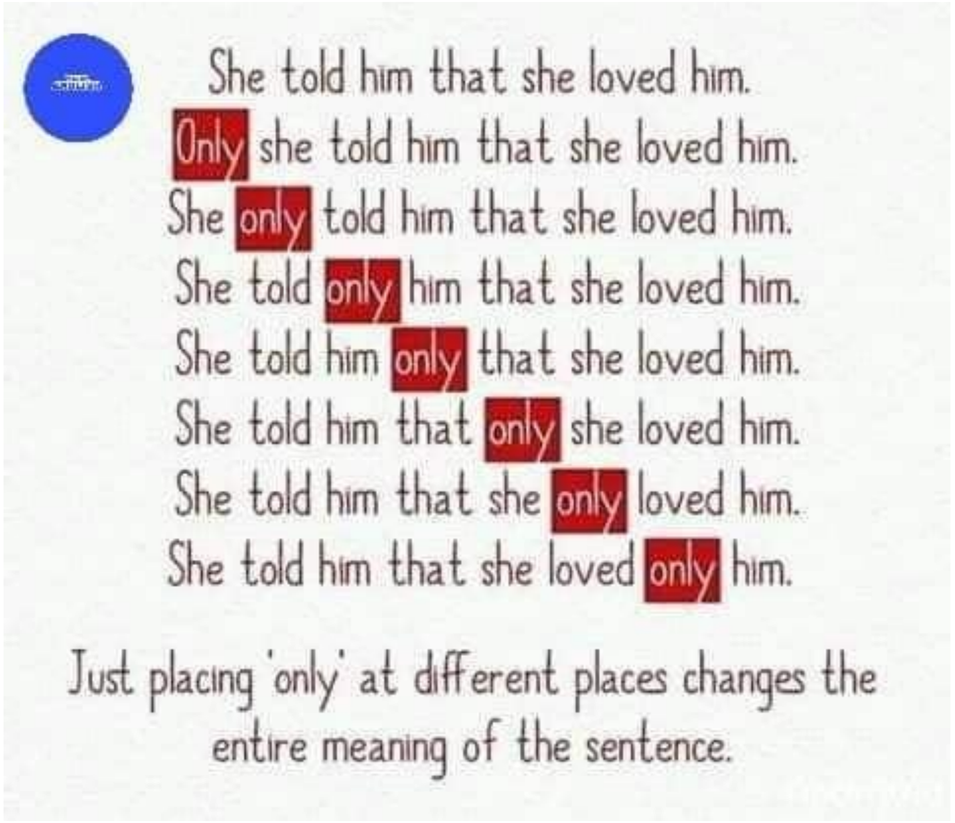
- Either a word is in a text or not (0 or 1), it cannot be 70% present.
    - is “Turku” in the sentence? → Yes/No
    - is “University” in the sentence? → Yes/No
    - is “university” in the lower-cased version of the sentence? → Yes/No

- Language itself is not discrete, but the way of representing it (text) is.

- The **order of words** in the sentence, also order of characters in the words are really important.
    - “CAT” is different from “TAC”

Farrokh Mehryary (farmeh@utu.fi)

NLP vs general machine learning



She told him that she loved him.
Only she told him that she loved him.
She only told him that she loved him.
She told only him that she loved him.
She told him only that she loved him.
She told him that only she loved him.
She told him that she only loved him.
She told him that she loved only him.

Just placing 'only' at different places changes the entire meaning of the sentence.

Image from The Language Nerds page in Facebook

NLP vs general machine learning

Set of words (also known as **bag of words** representation):

- [oman, syödä, valinnan, ihmistä, lahjakortilla, kaksi, tällä, mukaan, saa, listalta]

How would you reorder these words to form a sentence?

NLP vs general machine learning

Set of words (also known as **bag of words** representation):

- [oman, syödä, valinnan, ihmistä, lahjakortilla, kaksi, tällä, mukaan, saa, listalta]

How would you reorder these words to form a sentence?

1. “tällä lahjakortilla kaksi ihmistä saa syödä listalta oman valinnan mukaan” (with this gift card, two people are allowed to eat from the list of their choice)

Farrokh Mehryary (farmeh@utu.fi)

NLP vs general machine learning

Set of words (also known as **bag of words** representation):

- [oman, syödä, valinnan, ihmistä, lahjakortilla, kaksi, tällä, mukaan, saa, listalta]

How would you reorder these words to form a sentence?

1. “tällä lahjakortilla kaksi ihmistä saa syödä listalta oman valinnan mukaan” (with this gift card, two people are allowed to eat from the list of their choice)

2. “tällä lahjakortilla saa syödä kaksi ihmistä listalta oman valinnan mukaan” (with this gift card you can eat two people from the list of your choice)

NLP vs general machine learning

What determines the exact meaning of a sentence?

- The words in the sentence. (e.g., “*I ate a sandwich”* v.s “*I ate a burger*”).
- The grammatical structure of a sentence (syntax), determined by the order of its words.
- The context in which a sentence appears.

Farrokh Mehryary (farmeh@utu.fi)

NLP vs general machine learning

What determines the exact meaning of a sentence?

- The grammatical structure of a sentence (syntax), determined by the order of the words.

- In which order sentence elements usually appear?
  - Subject, predicate, object
- SVO: English, Finnish (e.g., Alex saw Riita.)
- SOV: Turkish, Japanese
- VSO: Wales, Arabic, Hebrew
- In theory, Finnish is a **free word order** language, however, in practice if you count the frequencies it’s SVO language
- In practice, sentences can be very complicated to understand.

NLP vs general machine learning

What determines the exact meaning of a sentence?

- The grammatical structure of a sentence (syntax), determined by the order of the words.

- **First step:**
  - Determining the type of each word in a sentence:

    (noun, verb, adjective, adverb, pronoun, preposition, conjunction, …)
  - This is called **part of speech tagging** (POS-tagging for short).

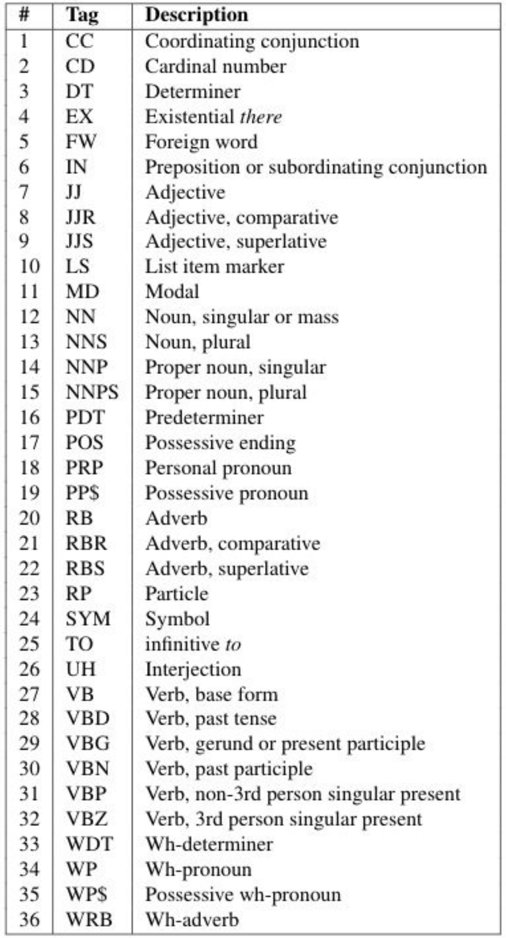
  - Some words have the same POS-tag, regardless of the context,
    - **globalization** is always a **noun**.

  - Some word can have different POS-tag in different contexts:
    - I **work** in the TurkuNLP group. → (verb)
    - My **work** is not easy. → (noun)

NLP vs general machine learning

What determines the exact meaning of a sentence?

- List of POS-tags for English



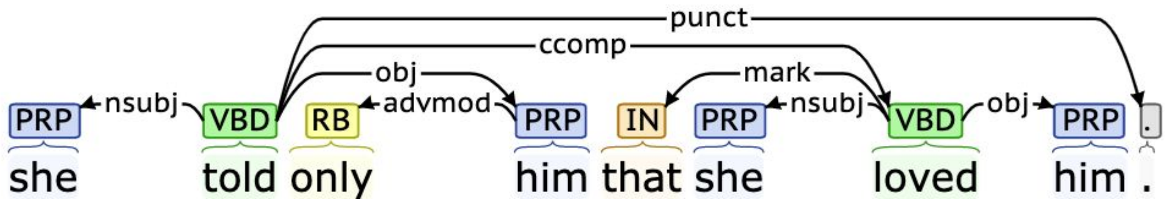
#	Tag	Description
1	CC	Coordinating conjunction
2	CD	Cardinal number
3	DT	Determiner
4	EX	Existential *there*
5	FW	Foreign word
6	IN	Preposition or subordinating conjunction
7	JJ	Adjective
8	JJR	Adjective, comparative
9	JJS	Adjective, superlative
10	LS	List item marker
11	MD	Modal
12	NN	Noun, singular or mass
13	NNS	Noun, plural
14	NNP	Proper noun, singular
15	NNPS	Proper noun, plural
16	PDT	Predeterminer
17	POS	Possessive ending
18	PRP	Personal pronoun
19	PP$	Possessive pronoun
20	RB	Adverb
21	RBR	Adverb, comparative
22	RBS	Adverb, superlative
23	RP	Particle
24	SYM	Symbol
25	TO	infinitive *to*
26	UH	Interjection
27	VB	Verb, base form
28	VBD	Verb, past tense
29	VBG	Verb, gerund or present participle
30	VBN	Verb, past participle
31	VBP	Verb, non-3rd person singular present
32	VBZ	Verb, 3rd person singular present
33	WDT	Wh-determiner
34	WP	Wh-pronoun
35	WP$	Possessive wh-pronoun
36	WRB	Wh-adverb

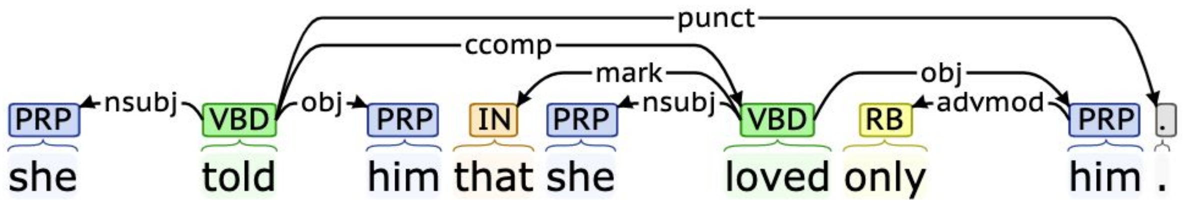
NLP vs general machine learning

What determines the exact meaning of a sentence?

- The grammatical structure of a sentence (syntax), determined by the order of the words.

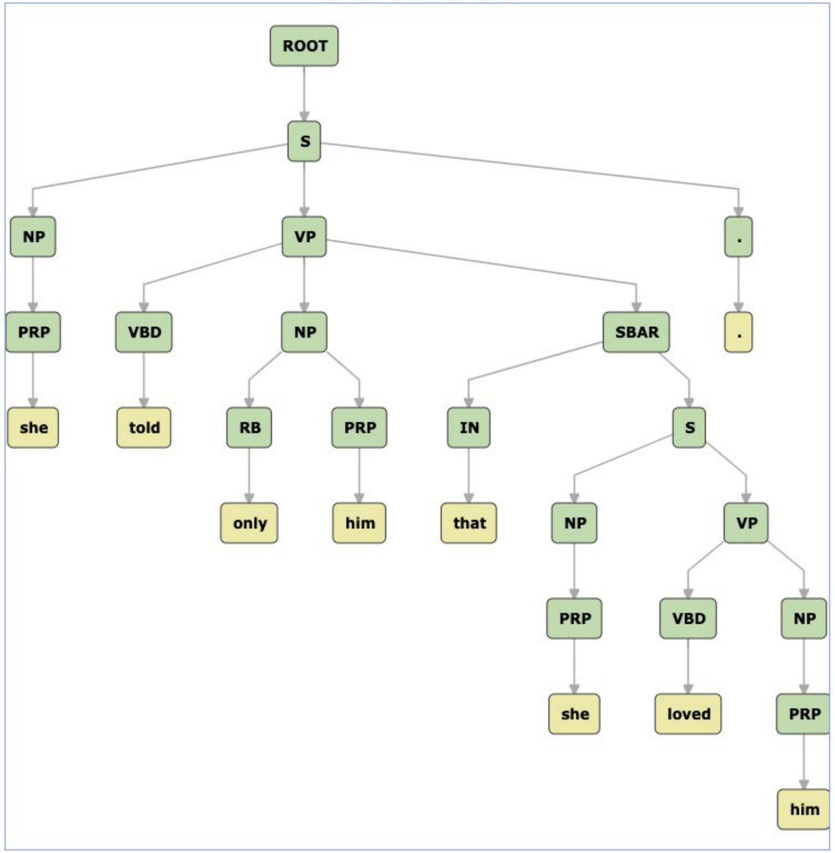
Second step: Analyzing the syntax of a sentence using i.e., dependency parsing.



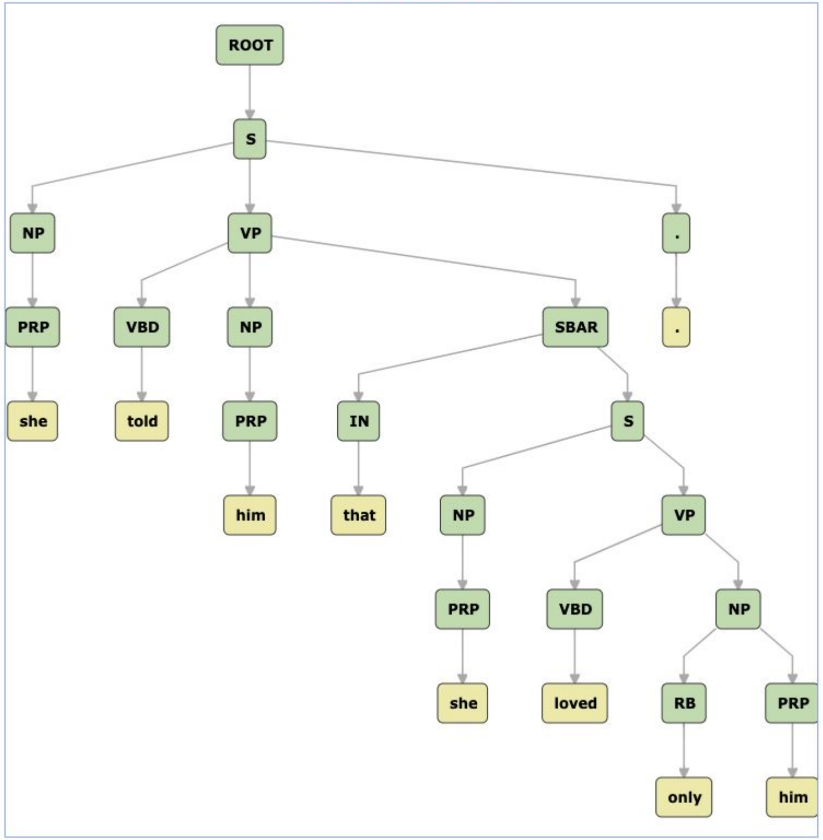


NLP vs general machine learning

Constituency Parse:



Constituency Parse:



NLP vs general machine learning

What determines the exact meaning of a sentence?

- The words in the sentence. (e.g., "*I ate a sandwich*" v.s "*I ate a burger*").
- The grammatical structure of a sentence (syntax), determined by the order of its words.
- **The context in which a sentence appears.**

Alex asked Ritta if he can leave her forever.

She laughed and replied 'i don't really mind!'

Alex asked if Ritta agrees to move in with him.

She laughed and replied 'i don't really mind!'

Farrokh Mehryary (farmeh@utu.fi)

NLP vs general machine learning

- We can see that representing the words in the sentence (e.g., as bag of words), determining the type of each word (POS-tagging), and analyzing the syntactic structure of the sentence (parsing), all can **extract useful information from the sentence** which can then be **used as features** in feature-based machine learning methods.
-
- Traditional NLP systems usually were implemented as **pipeline of different components**, each component **performing a specific task and/or extracting a set features**, and the last component in the pipeline used a machine learning model to perform the final task. Example:
    - Sentence boundary detection (Sentence segmentation)
    - Tokenization
    - POS-tagging
    - Parsing
    - Named-entity detection (finding names of people, cities, organizations, etc)
    - Relation extraction (who is the CEO of a company, who bought a company, etc).

- Modern deep learning based system only use tokenization, i.e., they do not rely on features such as POS-tags, dependency-types in the sentence parse graph, etc.

- End-to-end system: Raw data goes in, final predictions come out! (Being a pipeline or not).

NLP Systems:

1. Pre-processing and Features

Bag-of-words (no order): 1-gram

	w1	w2	w3	w4	w5	w6	w7	w8	...	wn
d1	0	0	1	0	0	0	1	0		0
d2	1	0	0	1	0	0	0	0		0
d3	0	0	1	1	0	0	1	0		1

Good for classification tasks:

- Email is Spam/not_spam
- News type: economy, sport, politics, military, etc
- Sentiment analysis: good movie, bad movie

NLP Systems:

1. Pre-processing and Features

Bag-of-words (no order): 1-gram

Problem 1:

- Cannot detect phrases, e.g., ‘University of Turku’.

Example: Find all documents talking about the University of Turku

- I work in the University of Turku. It has 25000 students and employees.
- A professor from the University of Helsinki visited Turku Cathedral today.

Solution: N-grams (N=2,3,4, …) , sliding window of length of N.

Three grams:

- I work in, work in the, in the University, University of Turku, ….
- A professor from, Professor from the, from the University, University of Helsinki, …

NLP Systems:

1. Pre-processing and Features

Bag-of-words (no order): 1-gram

Problem 2:

- Cannot detect how important a word is.

	w1	w2	w3	w4	w5	w6	w7	w8	...	wn
d1	4	8	1	0	0	0	2	0		0
d2	1	0	0	21	0	0	0	0		0
d3	0	0	0	1	0	0	3	0		3

Still problematic: because it is not suitable when documents have **different lengths**.
In practice, a news headline can be as important as an article. Also, it does not look at how frequent a term in the **whole corpus**.

NLP Systems:

1. Pre-processing and Features

Solution: TF-IDF or term frequency - inverse document frequency

```
t        : a term(e.g., a word)
d        : a document
tf(t,d)  : term frequently for term t in document d
df(t)    : document frequency
idf(t)   : inverse document frequency
```

tf(t,d) = count of t in d / number of words in d
df(t) = occurrence of t in N documents
idf(t) = N/df (is not suitable when N is very big like 10000, so we smooth it using log).
idf(t) = log(N/df)

→ tf-idf(t, d) = tf(t, d) * log(N/df)

NLP Systems:

1. Pre-processing and Features

Solution: TF-IDF or term frequency - inverse document frequency

→ tf-idf(t, d) = tf(t, d) * log(N/df)

Let's assume N=3

- word 'the' is in 3 documents → Log (3/3) =0
- word cat' is in 2 documents → Log (3/2) = 0.176
- word 'health' is in 1 document → log (3/1) = 0.477

-Term frequency shows how important is a term (word/character) in a particular document
-IDF shows how important a term is in whole corpus.

Farrokh Mehryary (farmeh@utu.fi)

NLP Systems:

1. Pre-processing and Features

Solution: Now instead of 0/1 or counts in the table, with use their corresponding tf-idf values

	w1	w2	w3	w4	w5	w6	w7	w8	...	wn
d1	v11	v12	v13	v14	v15	v16	v17	v18		v1n
d2	v21	v22	v23	v24	v25	v26	v27	v28		v29
d3	v31	v32	v33	v34	v35	v36	v37	v38		v39

In python use: `sklearn.feature_extraction.text.TfidfVectorizer`

When working with a big corpus (e.g., thousands of documents) and using for example 1,2,3,4,5 grams, this table can get huge, even having hundreds of thousands columns. Table will be sparse. We need ML algorithm that are proven to work well with such matrices (e.g., SVMs are very common in NLP).

NLP Systems:

1. Pre-processing and Features

Eliminating stop-words:

Stop words are a set of commonly used words in a language.
Examples of stop words in English are “a”, “the”, “is”, “are” and etc.
Stop words are commonly used in Text Mining and Natural Language Processing (NLP) to eliminate words that are so commonly used that they carry very little useful information.
Source: https://www.opinosis-analytics.com/knowledge-base/stop-words-explained/#.YVhe22YzZ60

- **In python use NLTK package, example here**: https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
- **Also:** TfidfVectorizer **can automatically remove stop words.**

- **Important:** never remove stop words before POS-tagging, sentence parsing, NER, relation extraction, summarization, translation, etc.
- **Only** useful for sentence and document classification/clustering tasks.

NLP Systems:

1. Pre-processing and Features

Text normalization: (optional, try and test if it can boost the performance)

1) **Text conversion:**
   a) Learn python **re** package for searching, replacing, removing and working with text data (e.g., re.sub)
   b) Learn python **bs4, BeautifulSoup, lxml** for HTML to clean text conversion
   c) Check **pdftotext** tool for PFD to clean text normalization

2) **Punctuation removal** (more text cleaning)
3) **Check/convert file encoding** (ASCII, UTF-8, etc)
4) **Lower-casing:** Lower-casing can work surprisingly well for classification/clustering task. (Be careful about abbreviations, and other NLP tasks such as named-entity recognition, relation extraction, etc).

NLP Systems:

1. Pre-processing and Features

Text normalization: (optional, try and test if it can boost the performance)

University, universities -> university
Played, playing, play, plays -> play
Activate, activates, activation, activated -> activate or activ

5) Stemming

- Stemming: a NLP pre-processing task which aims to reduce all morpholigical variants into a single stem form by trimming the suffixes. (**Activate, activates, activation, activated ➜** activ)
- It does not take into account the contexts of the words!
- It does not necessarily generate a valid dictionary form!
- Mistakes in stemming can appear:
    - Stemming can mistakenly reduce words with different meaning into the same stem. For example, “experiment” and “experience” could be both reduced into “experi” and “activates”, “activations” and “activities” could all be reduced into “activ” or “ act” bymost stemming algorithms
- **over/under stemming problem!**

NLP Systems:

1. Pre-processing and Features

Text normalization: (optional, try and test if it can boost the performance)

University, universities
Played, playing, play, plays
Activate, activates, activation, activated

6) Lemmatization (gives the dictionary form, considers the context, more accurate than stemming, usually based on ML methods, and relies on POS-tags and morphological tags).

Lemmatization is a NLP pre-processing task that converts a word (as it appears in a text) into its dictionary or base form, also known as **lemma**. Lemmatization transforms

- each **noun** into its **singular** form
- each **verb** into its **infinitive** form
- each **adjective/adverb** into its **base positive** form.

Examples:

- “sing”, “sings”, “sang” and “sung” are transformed into their common lemma “sing”
- “regulari**z**ation” and “regulari**s**ation” are both converted into “regularization”
- “am”, “is’ and “are” are converted into “be”
- “mouse” and “mice” are transformed into “mouse”.

NLP Systems:

1. Pre-processing and Features

Text normalization: (optional, try and test if it can boost the performance)

6) Lemmatization

Lemmatization example (relying on morphological and POS tags):

The word “**lives**” in English language can be either a verb (e.g. “He *lives* in Helsinki.“) or a plural noun (e.g. “This study examines the *lives* of the rich and famous.”), with the former being lemmatized into “live” and the latter into “life”.

- Check python **NLTK**, **Spacy**, and **NLTK.stem** packages for basic **sentence segmentation**, **tokenization**, and **stemming/lemmatization** algorithms.

- Check **UDPipe (https://ufal.mff.cuni.cz/udpipe)** software for advanced tokenization, sentence boundary detection, and lemmatization.

Farrokh Mehryary (farmeh@utu.fi)

NLP Systems:

2. NLP **pipelines** (End-to-end systems: raw text -> prediction)

- NLP systems are usually implemented as a pipeline: **raw texts -> system -> final predictions**

- Each component in the pipeline, does a pre-processing step and can either transform the text, or provide features that are needed for the subsequent components in the pipeline. The last component (or layer in a neural network) provides predictions.

Example: sentence/document classification, (is_spam/not_spam), (positive_review, negative_review, neutral_review, or detect types for the news articles: economy, sport, or **rank** a review from -10 to +10):

1) Text conversion (HTML, JSON, PDF, etc) -> text
2) Text clean-up
3) Sentence boundary detection (sentence segmentation) / tokenization
4) POS-tagging
5) Text normalization (lowe-casing and lemmatization)

6-1) Creating TF-IDF n-grams (bag of features for word-ngrams, lemma-ngrams, POS-tag-ngrams, etc) (**and other engineered features**) ➜ a Table ready for traditional for machine learning ➜ SVM training/prediction

6-2) Sequences of words/lemmas/POS-tags ➜ neural network ➜ neural network training/prediction

NLP Systems:

2. NLP **pipelines** (End-to-end systems: raw text -> prediction)

6-1) Creating TF-IDF n-grams (bag of features for word-ngrams, lemma-ngrams, POS-tag-ngrams, etc) (**and other engineered features**) ➔ a Table ready for traditional for machine learning ➔ SVM training/prediction

	w1	w2	w3	w4	w5	w6	w7	w8	...	wn
d1	v11	v12	v13	v14	v15	v16	v17	v18		v1n
d2	v21	v22	v23	v24	v25	v26	v27	v28		v29
d3	v31	v32	v33	v34	v35	v36	v37	v38		v39

- Creating a massive table of different features (either obtained by NLP pre-processing tasks or manually engineered features).
- Such tables are common NLP applications and are usually massive (thousands of columns, much less rows) and can be sparse.
- Use classifiers/regressors (two common algorithms SVMs and Random-forests).

NLP Systems:

2. NLP **pipelines** (End-to-end systems: raw text -> prediction)

6-2) Sequences of words/lemmas/POS-tags ➔ neural network ➔ neural network training/prediction

- **How to feed words, lemmas, or other features into neural networks? ➔ Embeddings:** word-embeddings, character-embeddings, POS-embeddings, lemma-embeddings, etc

- Each word (or lemma, POS, or other feature) is referenced by an index, and represented with a numerical vector (for example a 200-dimensional vector)

- Words that are more similar, will have similar vectors in the vector space.

- The embeddings can be **pre-trained in advance** (using unsupervised learning algorithms) and then plugged into neural networks, or **can be trained during the actual neural network training**.

- Sequence of vectors then can be fed into **RNNs, CNNs, and other types of neural networks**.

NLP Systems:

2. NLP **pipelines** (End-to-end systems: raw text -> prediction)

6-2) Sequences of words/lemmas/POS-tags → neural network → neural network training/prediction

- **How to feed words, lemmas, or other features into neural networks? → Embeddings:** word-embeddings, character-embeddings, POS-embeddings, lemma-embeddings, etc

Example:

A 200-d vector:

v1	v2	v3	v4	v5	v6	v7	v8	v9	...	v200

cat → index: 5464 → Vector of (cat)
dog → index: 7985 → Vector of (dog)
man → index: 2353 → Vector of (man)
woman → index: 2411 → Vector of (woman)

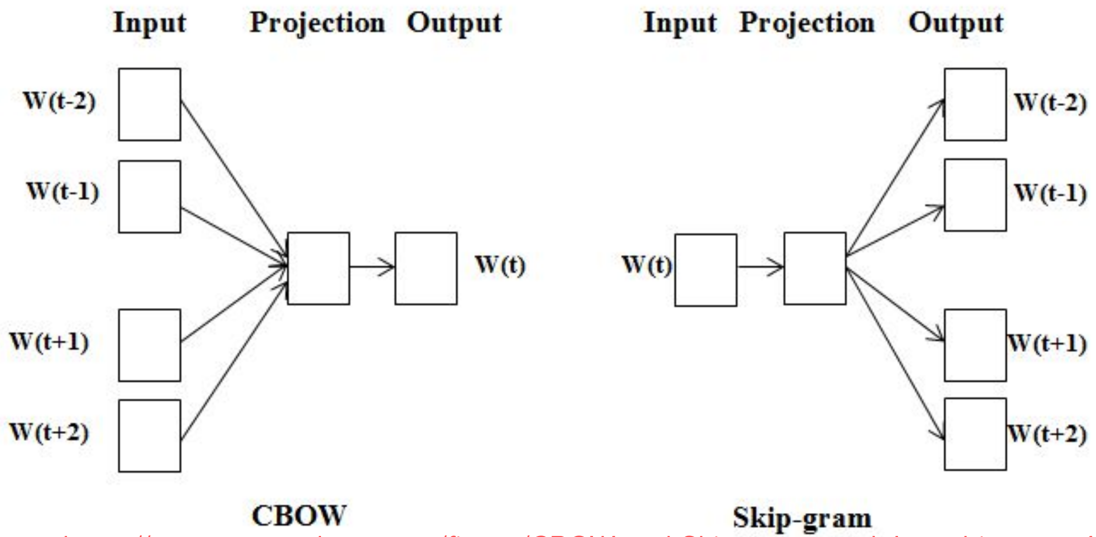
NLP Systems:

2. NLP **pipelines** (End-to-end systems: raw text -> prediction)

Word2vec algorithm: (pre-training word vectors on a massive corpus):

- **Hypothesis: similar words appear in similar contexts**

Sentence x: W1 W2 W3 W4 W5 … Wn , use sliding window with size = 5

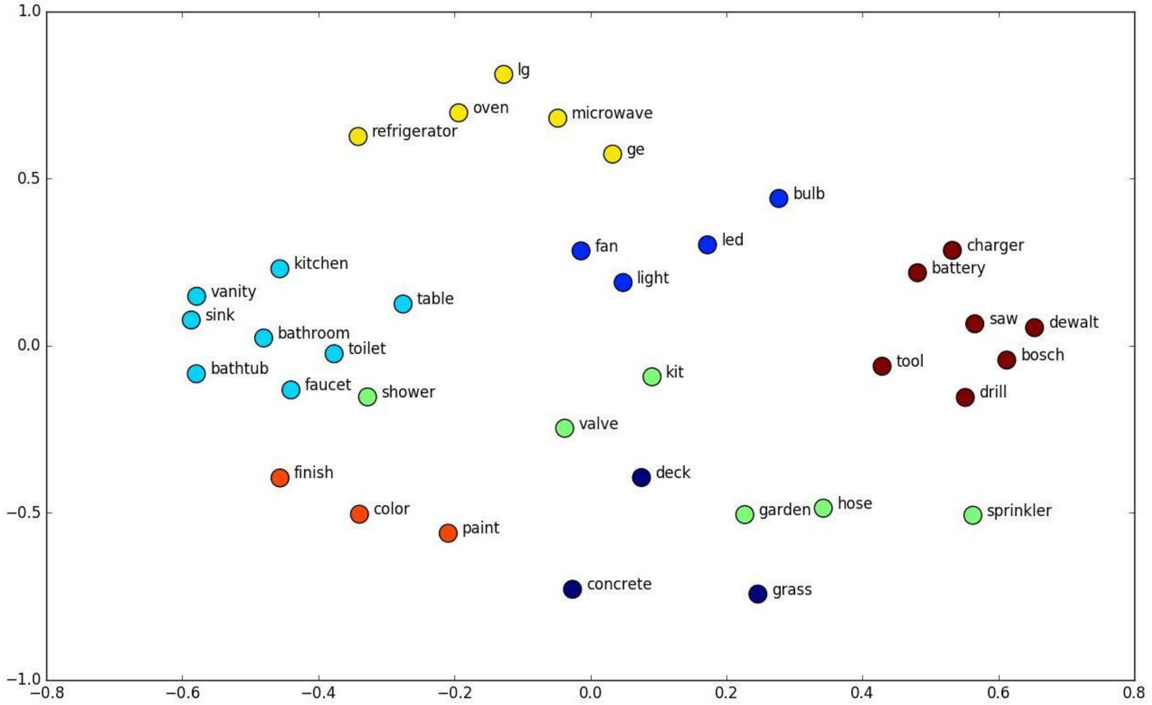


Source: https://www.researchgate.net/figure/CBOW-and-Skip-gram-models-architecture-1_fig1_319954363/download

NLP Systems:

2. NLP **pipelines** (End-to-end systems: raw text -> prediction)

Visualizing pre-trained words in a 2-d space using dimensionality reduction methods.

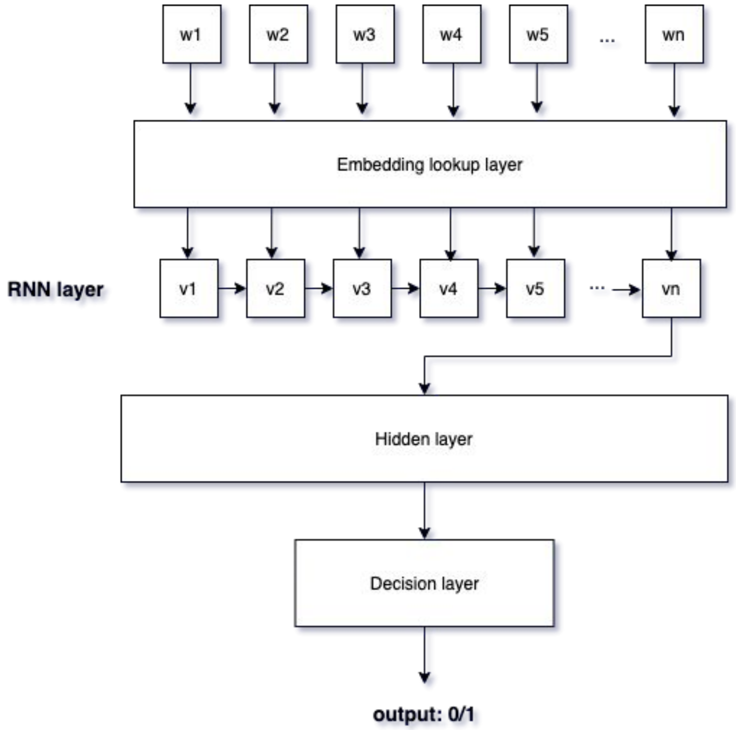


Source: https://www.shanelynn.ie/get-busy-with-word-embeddings-introduction/

NLP Systems:

2. NLP **pipelines** (End-to-end systems: raw text -> prediction)

Advanced example: Using word embeddings in neural network to detect is an email is spam or now



Using **keras** to define, train and evaluate a model.

```
model = Sequential()
model.add(Embedding(num_words, 200, input_length=maxlen))
model.add(LSTM(200,))
model.add(Dense(200, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam',
metrics = ['accuracy'])

model.fit(x_train, y_train, epochs=2, batch_size=256,
verbose=1)

scores = model.evaluate(x_test, y_test, verbose=1)
print("Accuracy: ", (scores[1]*100),"%")
```

NLP Tasks:

1- NLP preprocessing tasks

- Sentence boundary detection (sentence segmentation)
- Tokenization, sub-word tokenization
- Stemming / lemmatization
- POS-tagging
- Syntactic parsing, shallow parsing, etc
- ...

2-NLP tasks (downstream):

2-1: Unsupervised learning:

- Text clustering
- Topic modeling
- Information retrieval (e.g., search engines) , etc

2-2: Supervised learning tasks:

- Classification tasks
- Sequence tagging
- Sequence to sequence tasks
- Regression, Ranking, etc.

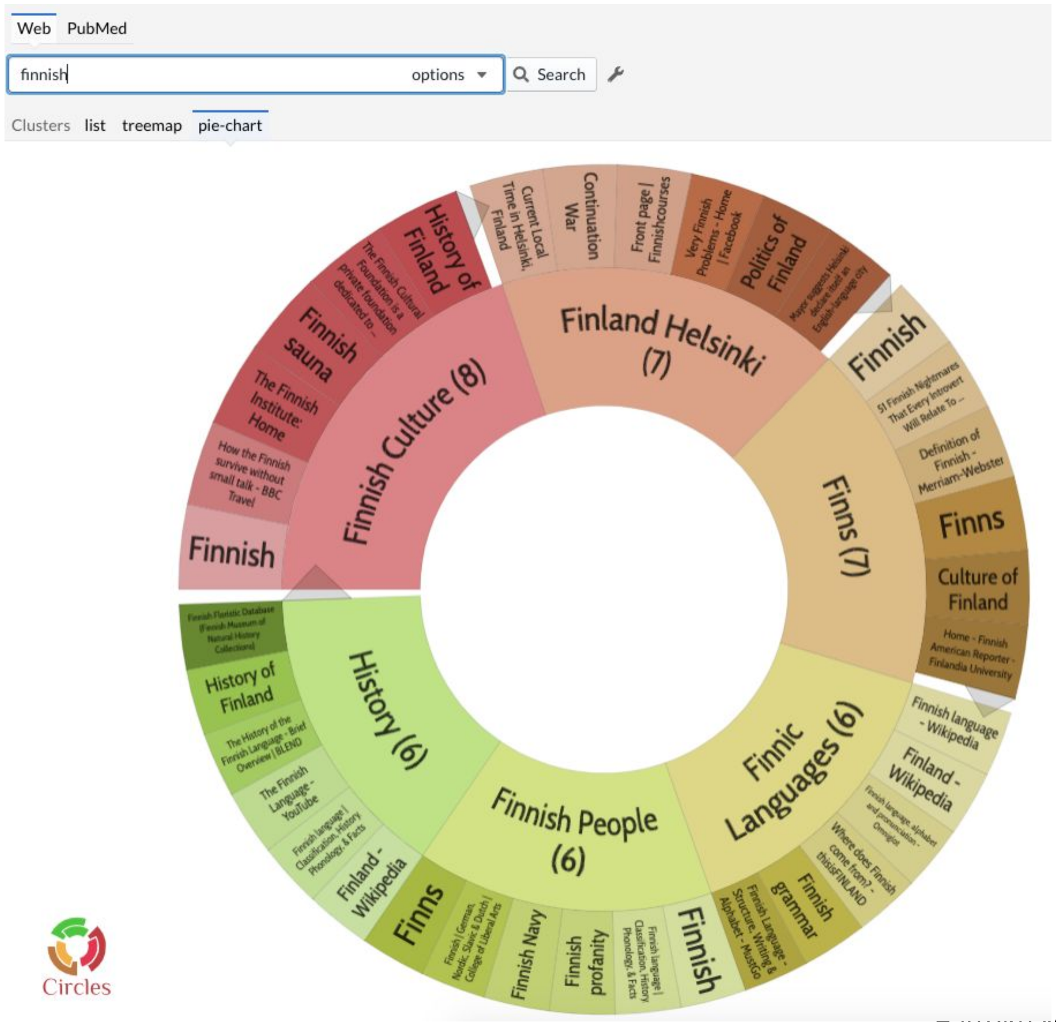
Farrokh Mehryary (farmeh@utu.fi)

NLP Tasks:

2-1: Unsupervised learning : **Clustering** example

Source: https://search.carrot2.org/

Hierarchical clustering of 100 top hits in google search.



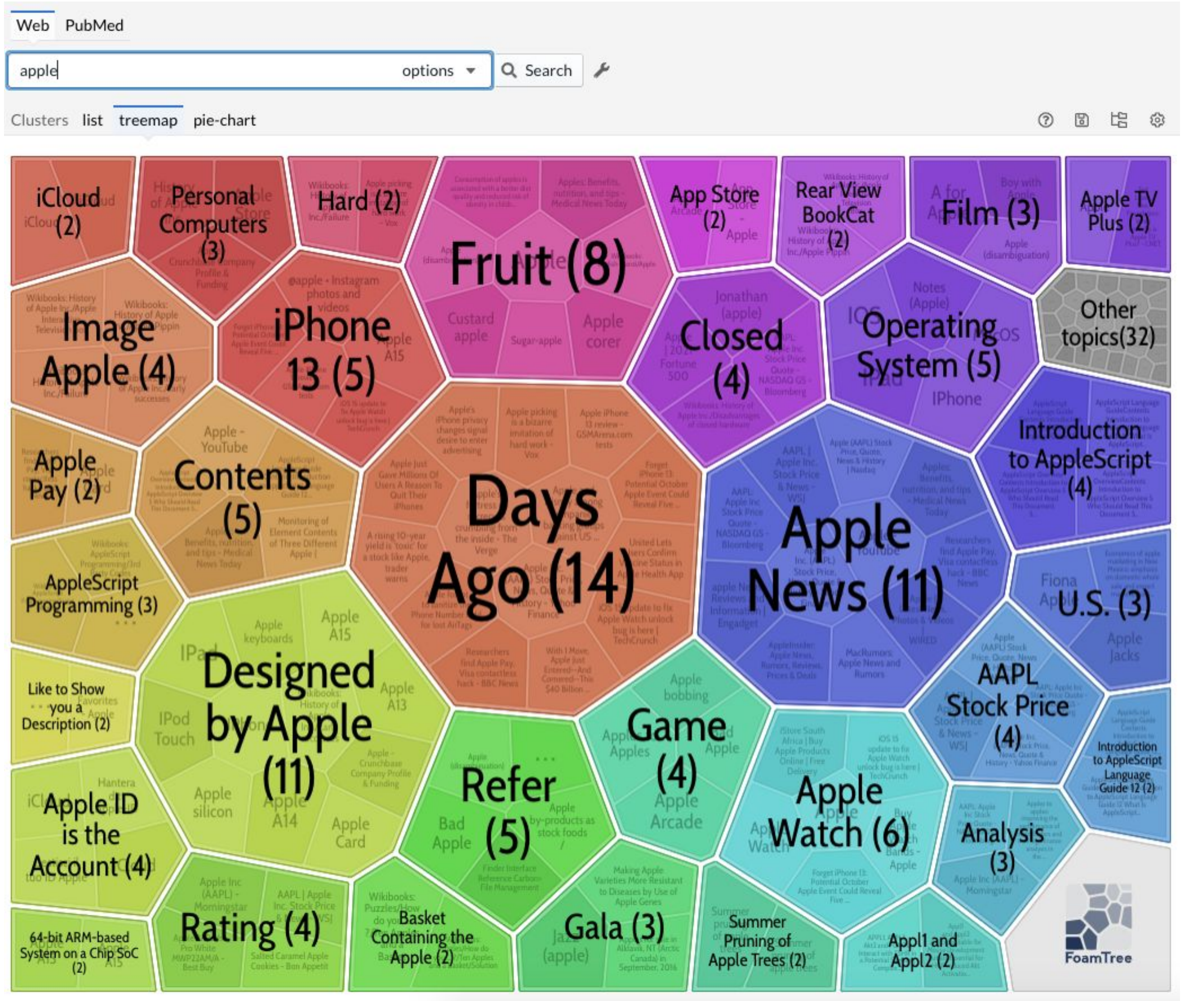
Web
PubMed
finnish
options
Search
Clusters
list
treemap
pie-chart
Finnish Culture (8)
Finland Helsinki (7)
Finns (7)
Finnic Languages (6)
Finnish People (6)
History (6)
History of Finland
The Finnish Cultural Foundation is a private foundation dedicated to ...
Finnish sauna
The Finnish Institute: Home
How the Finnish survive without small talk - BBC Travel
Finnish
Current Local Time in Helsinki, Finland
Continuation War
Front page | Finnishcourses
Very Finnish Problems - Home | Facebook
Politics of Finland
Mayor suggests Helsinki declare itself an English-language city
Finnish
51 Finnish Nightmares That Every Introvert Will Relate To ...
Definition of Finnish - Merriam-Webster
Finns
Culture of Finland
Home - Finnish American Reporter - Finlandia University
Finnish language - Wikipedia
Finland - Wikipedia
Finnish language, alphabet and pronunciation - Omniglot
Where does Finnish come from? - thisisFINLAND
Finnish grammar
Finnish Language - Structure, Writing & Alphabet - MustGo
Finnish
Finnish language | Classification, History, Phonology, & Facts
Finnish profanity
Finnish Navy
Finnish | German, Nordic, Slavic & Dutch | College of Liberal Arts
Finns
Finland - Wikipedia
Finnish language | Classification, History, Phonology, & Facts
The Finnish Language - YouTube
The History of the Finnish Language - Brief Overview | BLEND
History of Finland
Finnish Floristic Database (Finnish Museum of Natural History Collections)
Circles

NLP Tasks:

2-1: Unsupervised learning : **Clustering** example

Source: https://search.carrot2.org/

Hierarchical clustering of 100 top hits in google search.



NLP Tasks:

2-1: supervised learning:

2-1-1: Classification/regression tasks: Seq -> system -> Class label(s) or predicted value

1. Spam detection: Yes/No , Binary classification
2. Sentiment analysis: Good/Bad/Neutral review , Multi-class classification
3. Detect type of email or news: multi-label classification
4. Sentiment analysis: [-10, +10] (regression)
5. Relation extraction: can be binary/multi-class/multi-label classification
6. Other tasks ...

Sentiment analysis (opinion mining):

- search imdb review classification tutorial with python (TFIDFVectorizer, keras, tensorflow, pytorch, huggingface, etc).
- Can be about features (of products) in reviews: for example, when analyzing reviews about iphone 13, check what people say about **screen**, **battery**, **antena**, **camera**, etc.
- Very useful in business and politics.

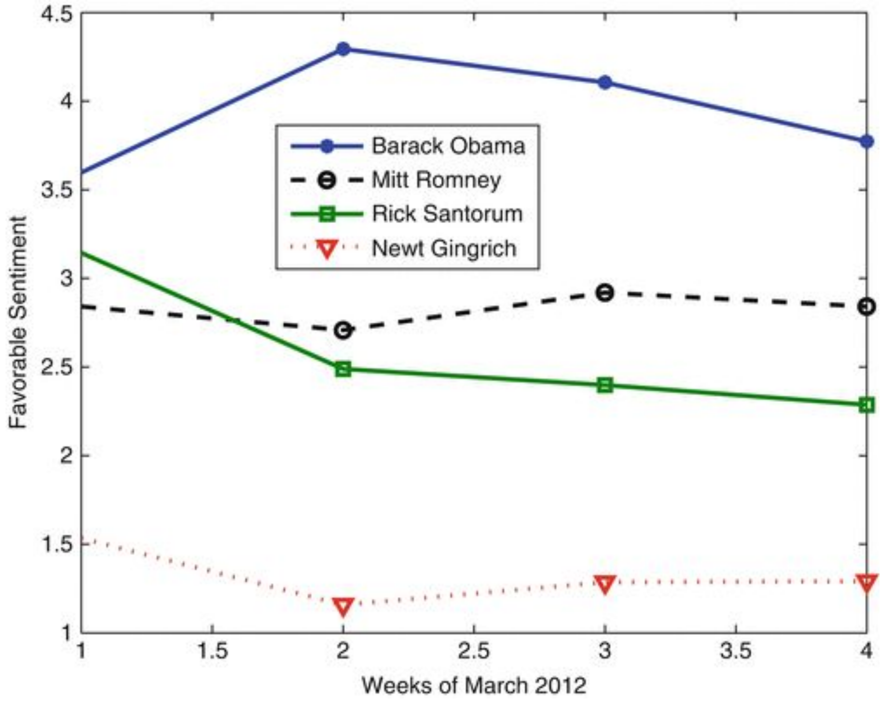
NLP Tasks:

2-1: supervised learning:
2-1-1: Classification/regression tasks: Seq -> system -> Class label(s) or predicted value

Sentiment analysis (opinion mining):

Example: mining twitter data about U.S candidates

Source:
https://link.springer.com/referenceworkentry/10.1007%2F978-1-4614-6170-8_351

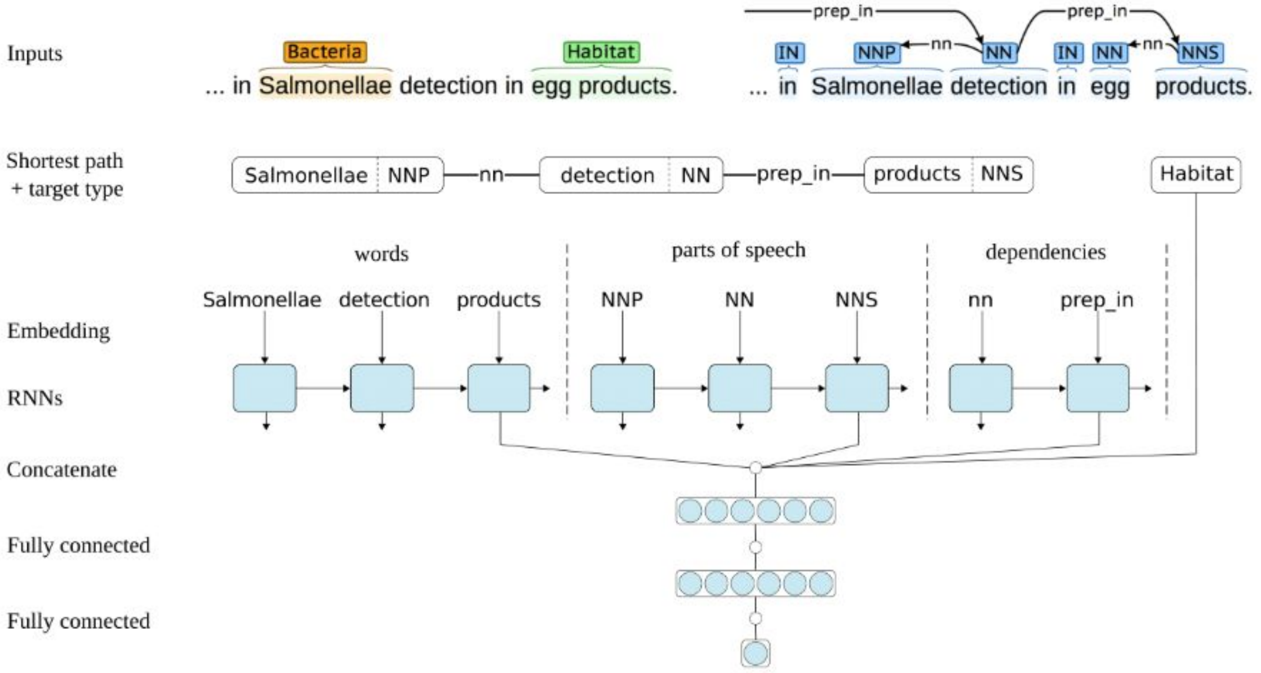


NLP Tasks:

2-1: supervised learning:
2-1-1: Classification/regression
Seq -> system -> Class label(s)

Relation extraction task:
Example:

If sentence says
a mentioned bactria,
lives_in a mentioned
habitat, or not?



Source: https://aclanthology.org/W16-3009/

NLP Tasks:

2-1: supervised learning:

2-1-2: Sequence tagging : seq of items -> system -> one tag (label) per item

- Usually characters or words
- Exactly one label per input item

Examples:

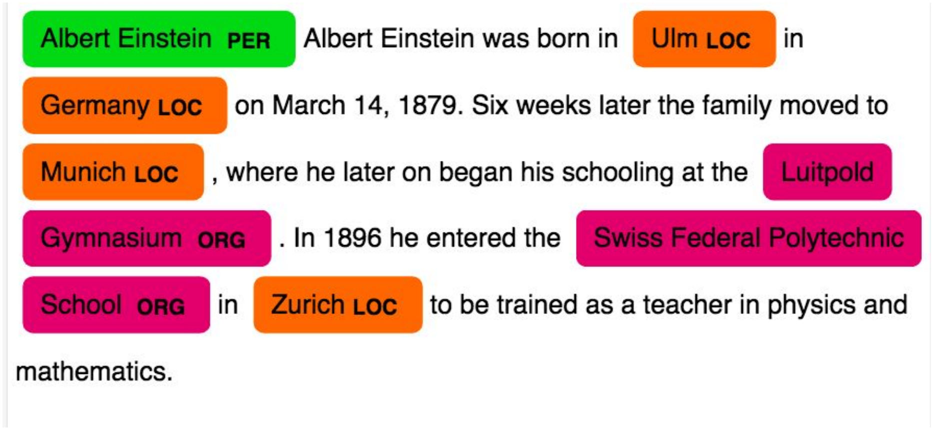
- Part-of-speech tagging (POS) -> gives the pos-tag of each word in the sentences.
- Named-entity recognition (NER) -> find mentions of locations, organizations, etc in texts.

NLP Tasks:

2-1: supervised learning:

2-1-2: Sequence tagging : seq of items -> system -> one tag (label) per item

- **Named-entity recognition (NER) examples:**



Albert Einstein PER Albert Einstein was born in Ulm LOC in Germany LOC on March 14, 1879. Six weeks later the family moved to Munich LOC , where he later on began his schooling at the Luitpold Gymnasium ORG . In 1896 he entered the Swiss Federal Polytechnic School ORG in Zurich LOC to be trained as a teacher in physics and mathematics.

Source: https://medium.com/@_alexgaus

NLP Tasks:

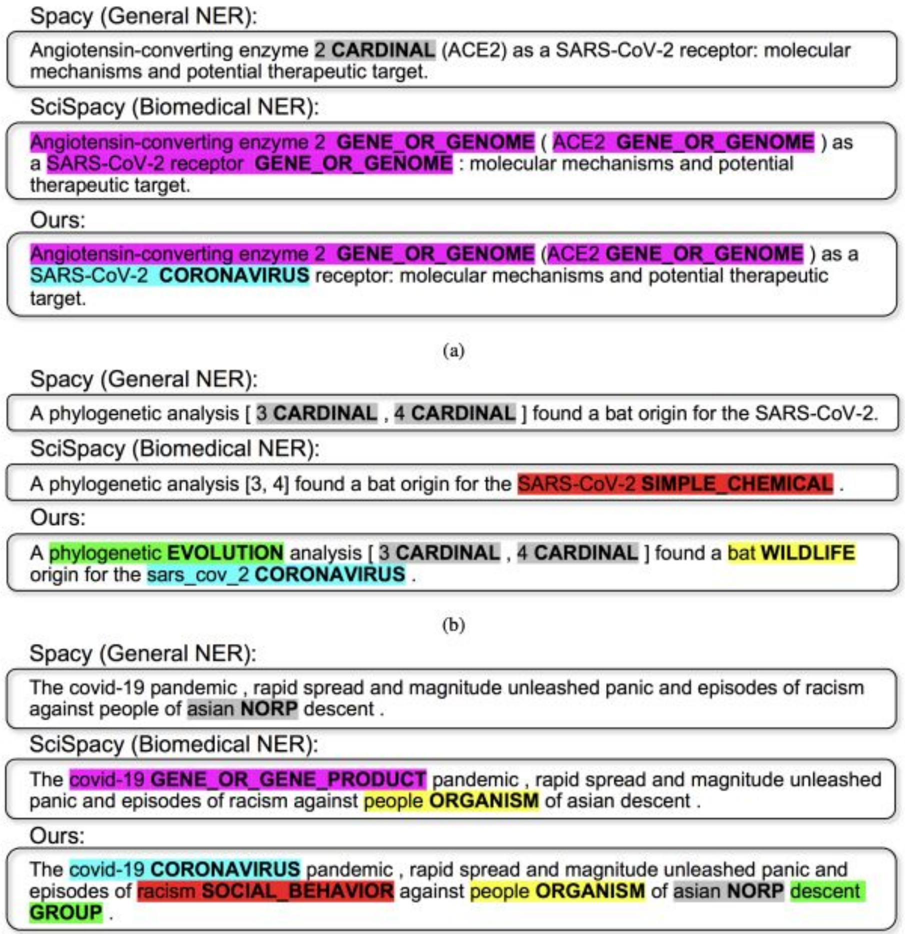
2-1: supervised learning:

2-1-2: Sequence tagging :

Named-entity recognition (NER) examples:

Biomedical NER example

Source: https://arxiv.org/pdf/2003.12218.pdf



Spacy (General NER):

Angiotensin-converting enzyme 2 **CARDINAL** (ACE2) as a SARS-CoV-2 receptor: molecular mechanisms and potential therapeutic target.

SciSpacy (Biomedical NER):

Angiotensin-converting enzyme 2 **GENE_OR_GENOME** (ACE2 **GENE_OR_GENOME**) as a SARS-CoV-2 receptor **GENE_OR_GENOME** : molecular mechanisms and potential therapeutic target.

Ours:

Angiotensin-converting enzyme 2 **GENE_OR_GENOME** (ACE2 **GENE_OR_GENOME**) as a SARS-CoV-2 **CORONAVIRUS** receptor: molecular mechanisms and potential therapeutic target.

(a)

Spacy (General NER):

A phylogenetic analysis [3 **CARDINAL** , 4 **CARDINAL**] found a bat origin for the SARS-CoV-2.

SciSpacy (Biomedical NER):

A phylogenetic analysis [3, 4] found a bat origin for the SARS-CoV-2 **SIMPLE_CHEMICAL** .

Ours:

A phylogenetic **EVOLUTION** analysis [3 **CARDINAL** , 4 **CARDINAL**] found a bat **WILDLIFE** origin for the sars_cov_2 **CORONAVIRUS** .

(b)

Spacy (General NER):

The covid-19 pandemic , rapid spread and magnitude unleashed panic and episodes of racism against people of asian **NORP** descent .

SciSpacy (Biomedical NER):

The covid-19 **GENE_OR_GENE_PRODUCT** pandemic , rapid spread and magnitude unleashed panic and episodes of racism against people **ORGANISM** of asian descent .

Ours:

The covid-19 **CORONAVIRUS** pandemic , rapid spread and magnitude unleashed panic and episodes of racism **SOCIAL_BEHAVIOR** against people **ORGANISM** of asian **NORP** descent **GROUP** .

NLP Tasks:

2-1: supervised learning:

2-1-3: Sequence to sequence tasks **: seq of items -> system -> sequence of items**

- Usually characters or words
- Length of the output sequence not limited!
- Predict a new item until the model predicts a stop-item

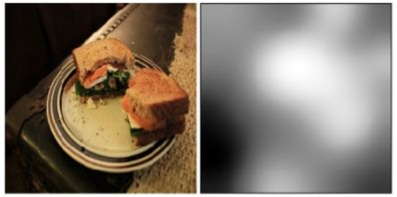
Examples:

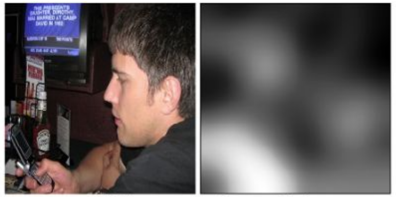
- Machine translation
- Summarization
- Data-to-text
- Caption generation
- Text generation (Poems, stories, fake news, etc ... Check GPT-2)

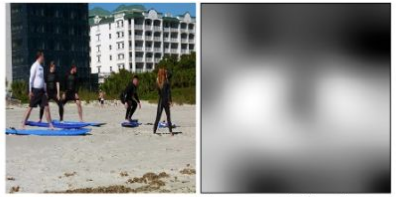
NLP Tasks:

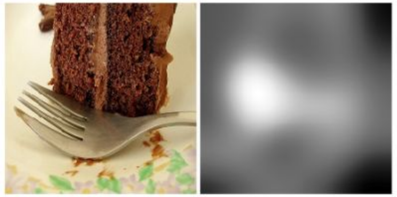
2-1: supervised learning: **image caption generation example**

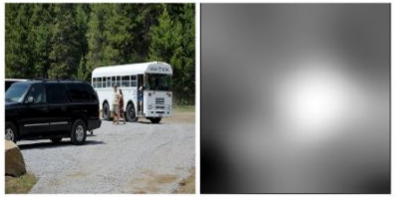
Source: here

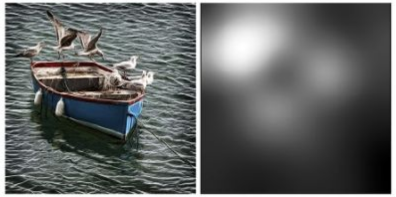

a sandwich sitting on top of a white plate
a sandwich cut in half on a plate
a sandwich is on a plate on a table

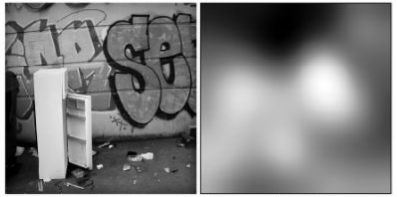

a man **sitting on a couch** on a cell phone
a man **holding** a cell phone **in his hand**
a man **sitting at a table** with a cell phone

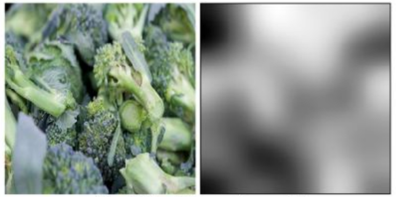

a group of people standing **around a beach**
a group of people standing **on a beach with surfboards**
a group of people standing **on top of a beach**

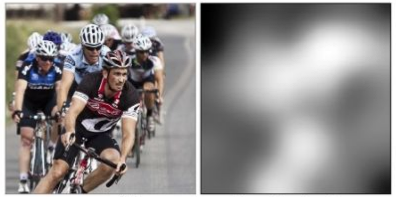

a piece of cake sitting **on top of a plate**
a piece of cake **on a plate with a fork**
a piece of chocolate cake **on a white plate**


a white passenger bus is parked on a street
a man standing next to a bus on a road
a bus is parked on the side of the road


a boat that is floating in the water
a group of birds sitting on top of a boat
two birds are sitting on a dock in the water


a toilet lying on its side on a sidewalk
an old refrigerator sitting in front of a wall
a old fashioned wall with **a clock** on it


a pile of lots of broccoli on a table
a close up of a bunch of broccoli
a close up of broccoli on a table


a man riding a motorcycle on the street
a group of people riding bikes down a street
a man riding a bike down a street

Further reading:

- NLP packages in python: NLTK and spaCy packages in python
- Stanford NLP tools (Stanford parser, etc). Demo: https://corenlp.run/
- Learn more about NLP: check courses offered by https://turkunlp.org/
- Learn more about deep learning: https://www.deeplearningbook.org/
- Deep learning packages in python: tensorflow, keras, torch (search for NLP/text mining examples)
- Check https://huggingface.co/ for advanced neural network models for NLP (BERT, GPT, GPT-2, GPT-3, T5, etc) : **Transformer-based models**.
- more examples: https://paperswithcode.com/datasets
- one more example: https://dropsofai.com/sentiment-analysis-with-python-tfidf-features/

Acknowledgement:

I would like to thank **Jenna Kanerva** for helping in a few slides and examples
(https://www.utu.fi/en/people/jenna-kanerva)