Data Types
&
Feature Engineering

Data Science & Machine Learning
ITS4 & ISE2

Mously DIAW
ENSAE - 2021

Summary

- Introduction
- Structured data
- Semi-structured data
- Unstructured data
- Questions

Data types

01

Tabular

That is, a table is followed by the names of the columns, followed by a sequence of rows.

Example of data:

- numeric
- categorical
- boolean
- Timestamp
- Compound data (list, set, tuple, dict, serialize, ...)

02

Image

Image classification (single-label)

Image classification (multi-label)

Image object detection

03

Text

Single / Multi label classification

Entity recognition

Sentiment Analysis

Text translation

04

Video

Action recognition

Classification

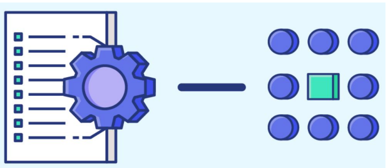
Object tracking

Video format:

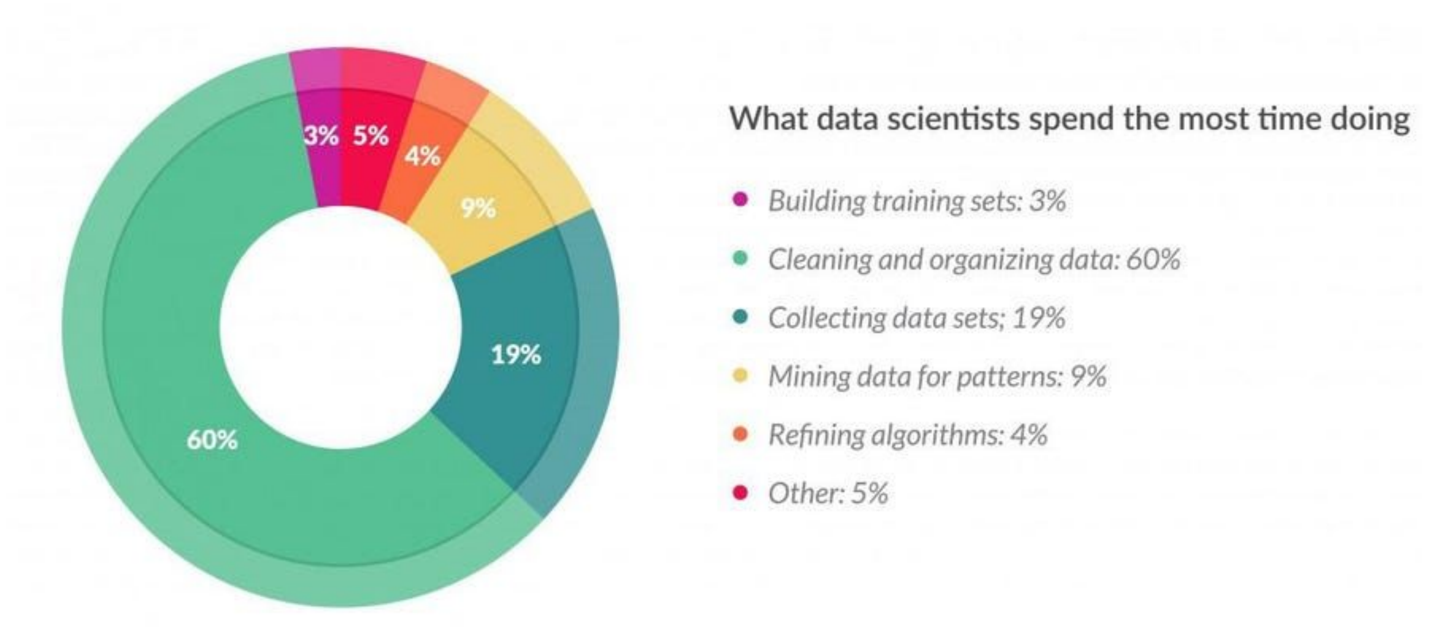
❏ .MOV
❏ .MPEG4
❏ .MP4
❏ .AVI

Feature Engineering

Feature engineering refers to the **process of using domain knowledge to select** and transform the most relevant variables from raw data when creating a predictive model using machine learning



Data preparation *accounts for about* **80%** *of the work of data scientists*



What data scientists spend the most time doing

- ● *Building training sets: 3%*
- ● *Cleaning and organizing data: 60%*
- ● *Collecting data sets; 19%*
- ● *Mining data for patterns: 9%*
- ● *Refining algorithms: 4%*
- ● *Other: 5%*

[Source: https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/](https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/)

Steps in Feature Engineering

01

Feature Extraction

Feature extraction is the automatic creation of new variables by extracting them from raw data

The purpose of this step is to automatically reduce the volume of data into a more manageable set for modeling

02

Data creation / preparation

Creating features involves identifying the variables that will be most useful in the predictive model

This is a subjective process that requires human intervention and creativity.

03

Exploratory Data Analysis (EDA)

This step is used to identify and summarize the main characteristics in a data set through data analysis and investigation

Data scientist determine which statistical techniques are most appropriate for data analysis, and for choosing the right features for a model.

04

Transformation / Selection

Transformation involves manipulating the predictor variables to improve model performance

Feature selection algorithms essentially analyze, judge, and rank various features to determine which features are irrelevant/redundant and should be removed

Feature Engineering process

- Data preparation and exploratory data analysis
- Brainstorming/testing features and choosing which features to create
- Creating features
- Testing the impact of the identified features on the task;
- Optimizing the features if needed and repeating until the features work effectively

Typical engineered features

The following list provides some typical ways to engineer useful features

- Numerical transformations (like scaling)
- Category encoder like one-hot encoder (for categorical data)
- Clustering
- Group aggregated values
- Principal component analysis (PCA for numerical data)
- Data imputation handling outliers

NLP

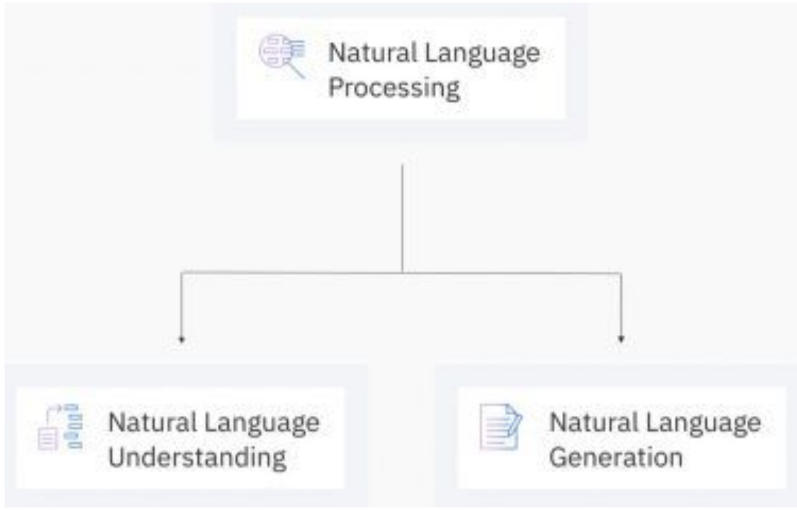
Natural Language Processing

What is NLP ?

Natural language processing (NLP) refers to the branch of computer science concerned with **giving computers the ability to understand text and spoken words in much the same way human beings can**. NLP seeks to convert unstructured language data into a structured data format to enable machines to understand speech and text and formulate relevant, contextual responses

Natural language understanding (NLU) uses **syntactic and semantic analysis of text and speech to determine the meaning** of a sentence. NLU focuses on machine reading comprehension through grammar and context, enabling it to determine the intended meaning of a sentence.

Natural language generation (NLG) is the process of **producing a human language text response based on some data input.** NLG enables computers to write. NLG focuses on text generation, or the construction of text in English or other languages, by a machine and based on a given dataset.


Natural Language Processing
Natural Language Understanding
Natural Language Generation

NLP tasks

Speech recognition, also called speech-to-text, is the task of reliably converting voice data into text data.

Named entity recognition, or NEM, identifies words or phrases as useful entities. NEM identifies ‘Kentucky’ as a location or ‘Fred’ as a man's name.

Part of speech tagging, also called grammatical tagging, is the process of determining the part of speech of a particular word or piece of text based on its use and context. Part of speech identifies ‘make’ as a verb in ‘I can make a paper plane,’ and as a noun in ‘What make of car do you own?’

Sentiment analysis attempts to extract subjective qualities-attitudes, emotions, sarcasm, confusion, suspicion-from text.

Natural language generation (NLG) is sometimes described as the opposite of speech recognition or speech-to-text; it's the task of putting structured information into human language.

Word sense disambiguation is the selection of the meaning of a word with multiple meanings through a process of semantic analysis that determine the word that makes the most sense in the given context. For example, word sense disambiguation helps distinguish the meaning of the verb 'make' in ‘make the grade’ (achieve) vs. ‘make a bet’ (place).

Co-reference resolution is the task of identifying if and when two words refer to the same entity. The most common example is determining the person or object to which a certain pronoun refers (e.g., ‘she’ = ‘Mary’),

NLP tools and approaches

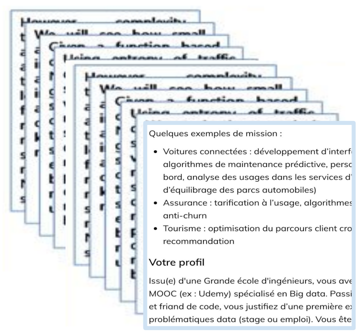
rules-based (RB) systems could perform certain NLP tasks, but couldn't easily scale to accommodate a seemingly endless stream of exceptions or the increasing volumes of text and voice data.

statistical NLP combines computer algorithms with machine learning and deep learning models to automatically extract, classify, and label elements of text and voice data

NLP use cases:

Spam detection (gmail)
Machine translation (Google translate, Deepl)
Virtual agents and chatbots (Alexa, Siri, virtual assistant)
Social media sentiment analysis (social media post, tweets, ecommerce reviews, ...)
Text summarization (research databases, article keywords)

ML algorithm processing only **numerical data**


Quelques exemples de mission :

- Voitures connectées : développement d'interf
algorithmes de maintenance prédictive, persc
bord, analyse des usages dans les services d'
d'équilibrage des parcs automobiles)
- Assurance : tarification à l'usage, algorithmes
anti-churn
- Tourisme : optimisation du parcours client cro
recommandation

Votre profil

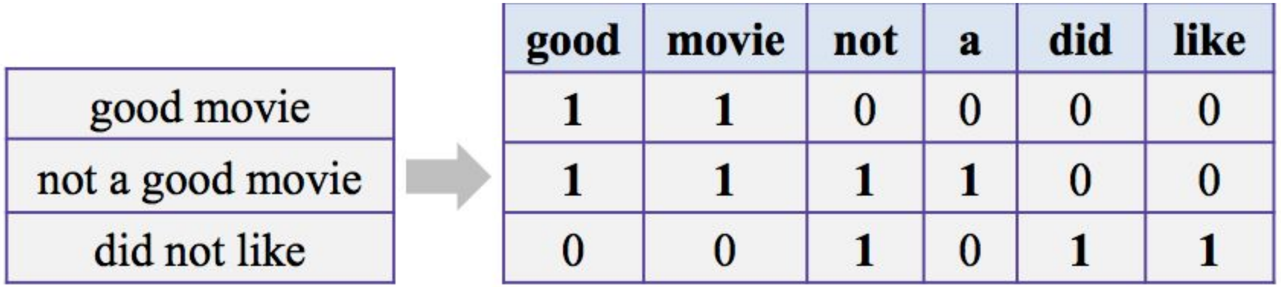
Issu(e) d'une Grande école d'ingénieurs, vous ave
MOOC (ex : Udemy) spécialisé en Big data. Passi
et friand de code, vous justifiez d'une première ex
problématiques data (stage ou emploi). Vous ête

	t_1	t_2	t_3	...	t_n
D_1	$tfidf_{1,1,N}$	$tfidf_{2,1,N}$	$tfidf_{3,1,N}$	...	$tfidf_{n,1,N}$
D_2	$tfidf_{1,2,N}$	$tfidf_{2,2,N}$	$tfidf_{3,2,N}$	...	$tfidf_{n,2,N}$
...	...	...	...	...	...
D_N	$tfidf_{1,N,N}$	$tfidf_{2,N,N}$	$tfidf_{3,N,N}$	...	$tfidf_{n,N,N}$

Terminologies

- **Document:** A document is a single text data point. For Example, a review of a particular product by the user.
- **Corpus:** It a collection of all the documents present in our dataset.
- **Feature**: every unique token (word or sequence of words) in the corpus is considered as a feature.

Binary vectorizer (bag of words)



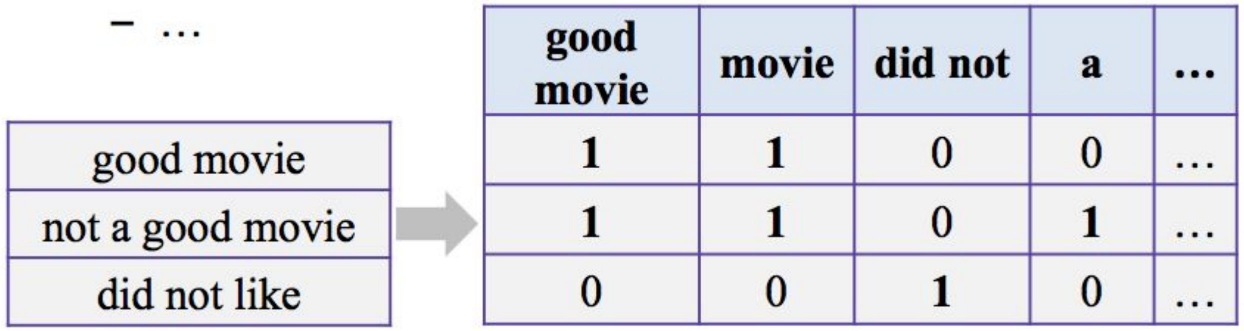
	good	movie	not	a	did	like
good movie	1	1	0	0	0	0
not a good movie	1	1	1	1	0	0
did not like	0	0	1	0	1	1

It is obvious that it has some problems:

- We lose word order, hence the name “bag of words” as they're not ordered
- The counters are not normalized.

Binary vectorizer (n-grams)

Example: **2-grams or bi-grams**

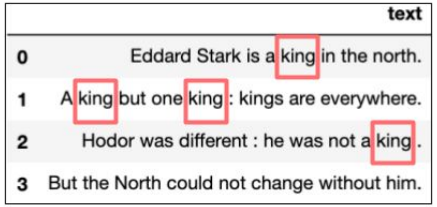


good movie
not a good movie
did not like

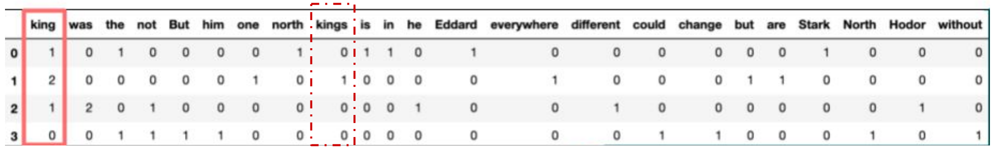
good movie	movie	did not	a	...
1	1	0	0	...
1	1	0	1	...
0	0	1	0	...

- The counters are not normalized
- Sparse matrix (contain mostly zero values)

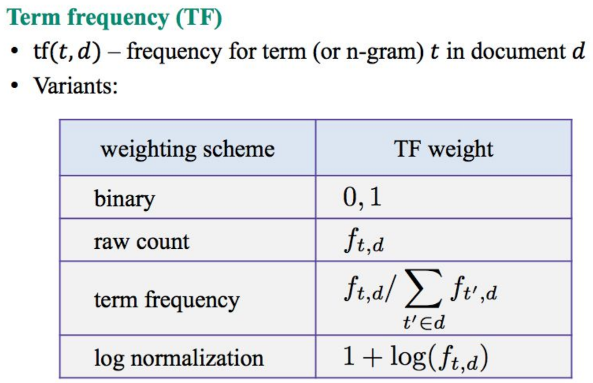
Term Frequency (TF) or Count vectorization



	text
0	Eddard Stark is a king in the north.
1	A king but one king : kings are everywhere.
2	Hodor was different : he was not a king .
3	But the North could not change without him.



	king	was	the	not	But	him	one	north	kings	is	in	he	Eddard	everywhere	different	could	change	but	are	Stark	North	Hodor	without
0	1	0	1	0	0	0	0	1	0	1	1	0	1	0	0	0	0	0	0	1	0	0	0
1	2	0	0	0	0	0	1	0	1	0	0	0	0	1	0	0	0	1	1	0	0	0	0
2	1	2	0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0
3	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1



Term frequency (TF)

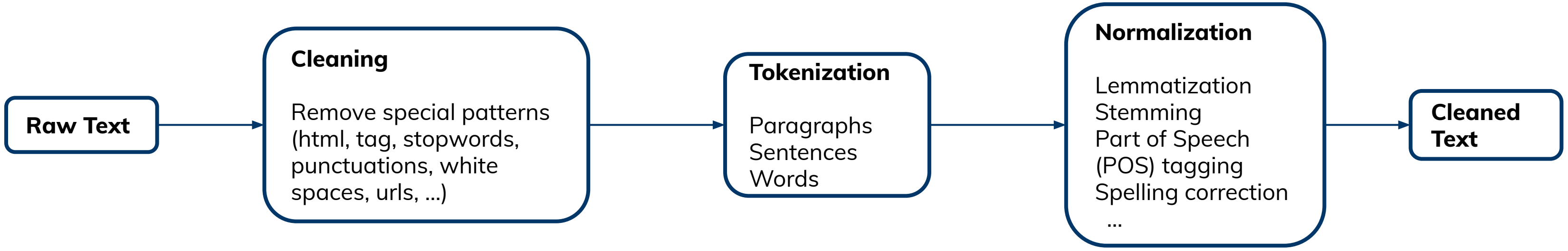
- $\mathrm{tf}(t, d)$ – frequency for term (or n-gram) t in document d
- Variants:

weighting scheme	TF weight
binary	$0, 1$
raw count	$f_{t,d}$
term frequency	$f_{t,d} / \sum_{t' \in d} f_{t',d}$
log normalization	$1 + \log(f_{t,d})$

Drawbacks:

- The counters are not normalized
- Sparse matrix
- Terms are usually not statistically independent
- Missing semantic sensitivity

What is NLP text preprocessing?



Cleaning step assumes removing all undesirable content.

Tokenization - text preprocessing step, which assumes splitting text into tokens(words, sentences, etc.)

Normalization is a conversion of any non-text information into textual equivalent

Converting dates to text
Numbers to text
Currency/Percent signs to text
Spelling mistakes correction

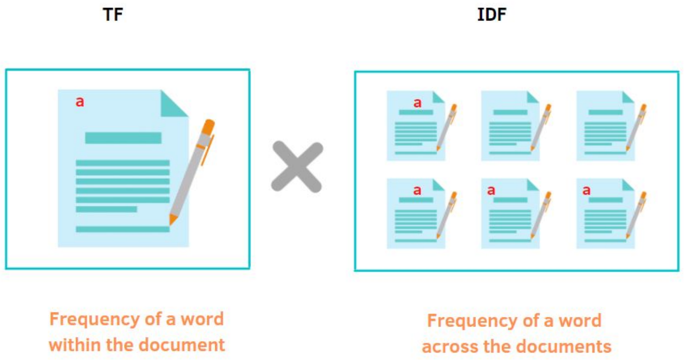
Notes

Stemming is the process of reducing inflection in words to their root forms such as mapping a group of words to the same stem even if the stem itself is not a valid word in the Language (ex: **studies => studi**)

Lemmatization, unlike Stemming, reduces the inflected words properly ensuring that the root word belongs to the language.Lemmatization considers the context and converts the word to its meaningful base form, which is called lemma. (ex: **studies => study**)

TF-IDF

Term frequency - inverse document frequency


TF
IDF
a
Frequency of a word within the document
Frequency of a word across the documents

Inverse document frequency (IDF)

- $N = |D|$ – total number of documents in corpus
- $|\{d \in D : t \in d\}|$ – number of documents where the term t appears
- $\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$

TF-IDF

- $\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$
- A high weight in TF-IDF is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents

Drawbacks:

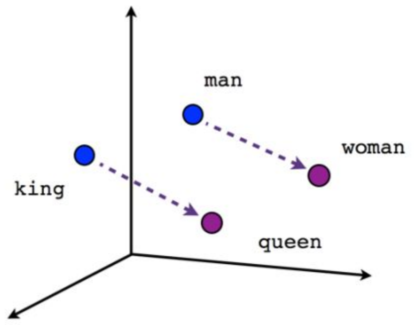
- Sparse matrix, scalability (Very high dimensional)
- Low/High frequency
- Untreated synonym
- Terms are usually not statistically independent
- Missing semantic sensitivity

Word embedding

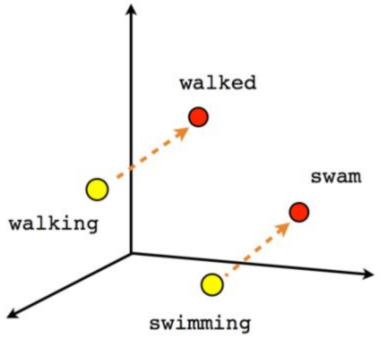
Word embedding is a term used for the representation of words for text analysis, typically in the form of a real-valued vector that encodes the meaning of the word

Word embedding is able to capture the context, the semantic and syntactic similarity (gender, synonyms, ...) of a word by reducing the dimension. Word2Vec is the first neural embedding model

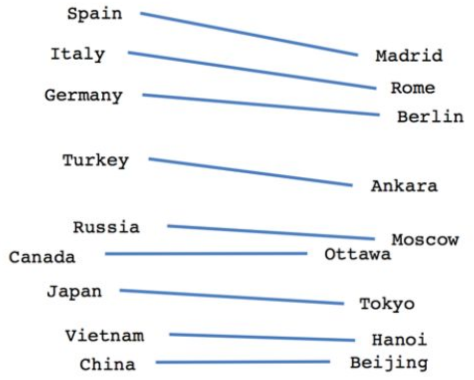
Methods to generate this mapping include neural networks, dimensionality reduction on the word co-occurrence matrix



Male-Female



Verb tense

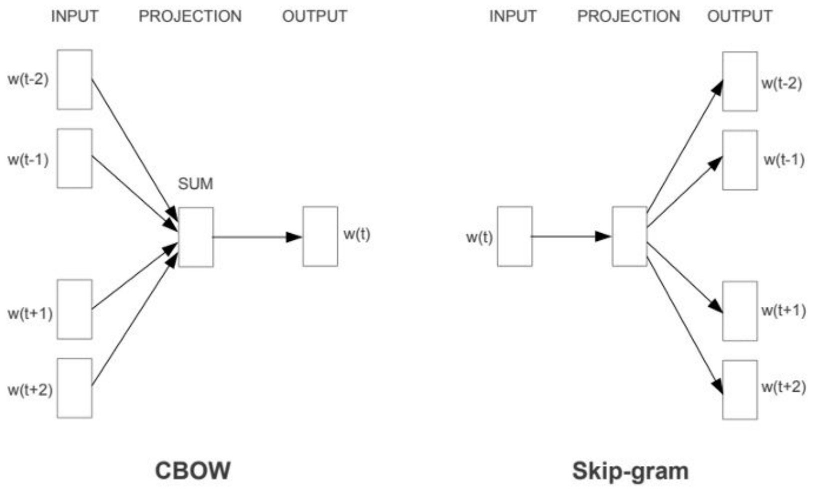


Country-Capital

Word embedding

Word2Vec

Word2Vec is a statistical method for efficiently learning a standalone word embedding from a text corpus. It was developed by Tomas Mikolov, et al. at Google in 2013



Continuous Bowl of Words(CBOW) try to fit the neighboring words in the window to the central word.

Skip Gram try to make the central word closer to the neighboring words. It is the complete opposite of the CBOW model

Train your own word2vec model on a **custom corpus**

Pre-trained Word Embedding Models:

- Word2Vec (Google)
- GloVe (Stanford, Global Vectors for Word Representation)
- SpaCy
- fastText (Facebook, Enriching Word Vectors with Subword Information)
- Flair (University of Berlin) etc.

Word embedding

In very simple terms, Word Embeddings are the texts converted into numbers and there may be different numerical representations of the same text.

Benefits of using Word Embeddings:

- It is much faster to train than hand build models like WordNet(which uses ***graph embeddings***)
- Almost all modern NLP applications start with an embedding layer
- It Stores an approximation of meaning
- dense and low-dimensional vectors is computational

Drawbacks of Word Embeddings:

- It can be memory intensive
- It is corpus dependent. Any underlying bias will have an effect on your model
- It cannot distinguish between homophones. Eg: brake/break, cell/sell, weather/whether etc.

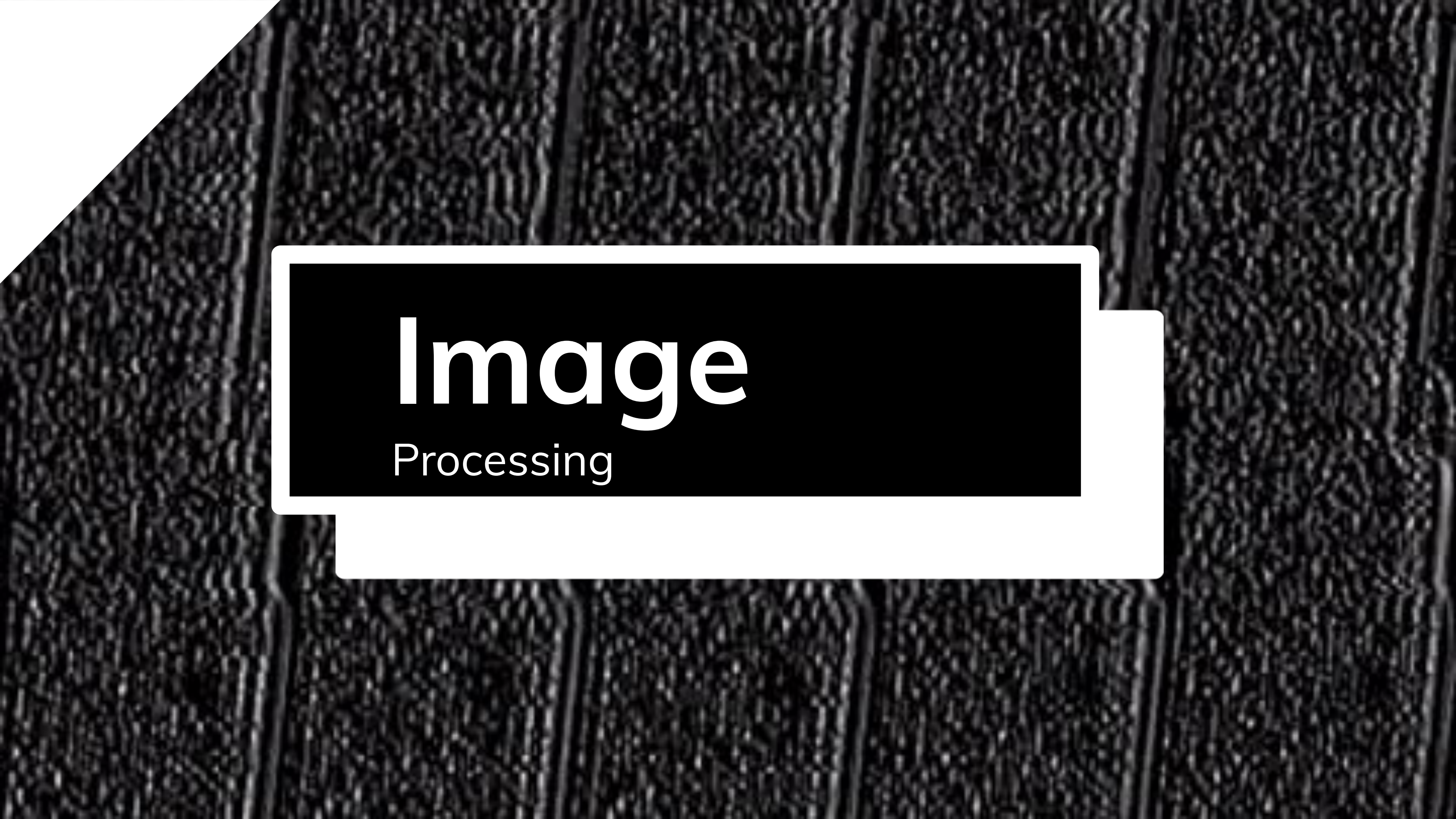
Word Embedding (CNN, RNN, ELMo, ...)

A proliferation of NLP programs, especially among the digital giants. Among the most advanced models, we can mention:

Google: **Bert, Albert**

- **Derived models**
  - RoBERTa (Facebook)
  - StructBERT (Alibaba)
  - DeBERTa (Microsoft)
  - DistilBERT (Hugging Face)

- **Alternative models**
  - GPT-2 et GPT-3 (OpenAI)
  - XLNet (Université Carnegie Mellon)
  - UniLM (Microsoft)
  - Reformer (Google)



Image

Processing

What is image processing?

Image preprocessing is one of the most under-explored problems in the data science community. Image preprocessing converts image data into a form that allows machine learning algorithms to solve it. Some of the tools and platforms used in image preprocessing include: Pytorch, OpenCV, Keras, Tensorflow, and Pillow. Example of use cases:

- Face detection
- Medical imaging
- Machine vision
- Autonomous Driving
- Image matching

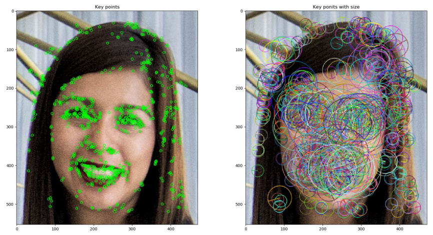


Image processing flowchart

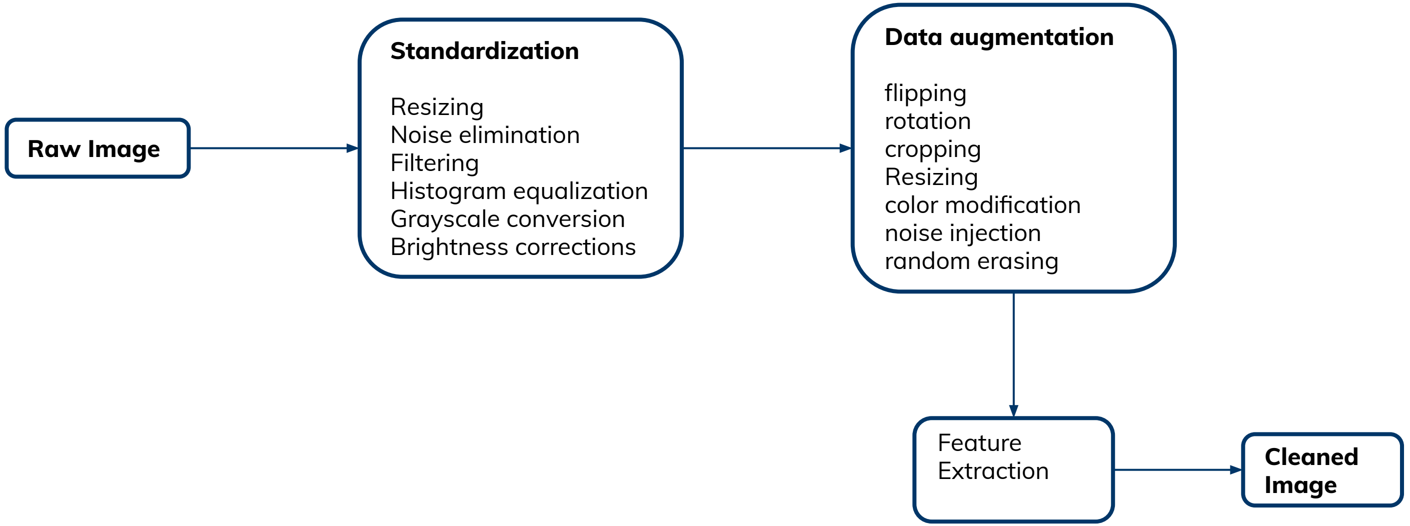


Image processing - Data augmentation

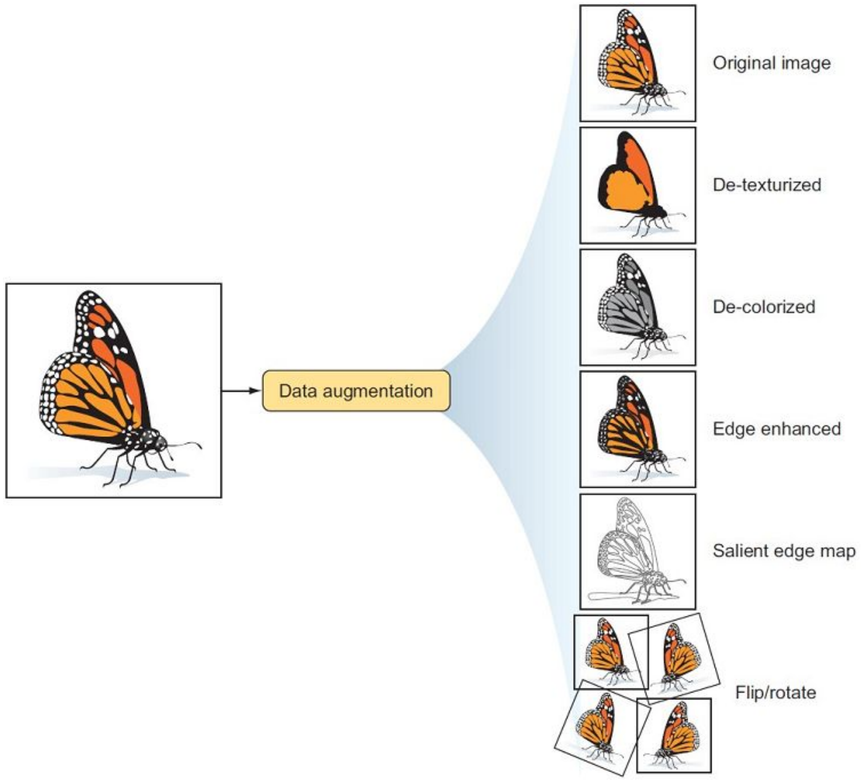


Image processing - Feature Extraction

Algorithms for Identification

- Harris Corner
- SIFT(Scale Invariant Feature Transform)
- SURF(Speeded Up Robust Feature)
- FAST(Features from Accelerated Segment Test)
- ORB(Oriented FAST and Rotated BRIEF)
- BRIEF (Binary Robust Independent Elementary Features)

Best practices

Best practices for creating training data

➔ **Avoid target leakage**

Target leakage happens when your **training data includes predictive information that is not available when you ask for a prediction.** Target leakage can cause your model to show excellent evaluation metrics, but perform poorly on real data.

➔ **Avoid training-serving skew**

Training-serving skew happens when you generate your **training data differently** than you generate the data you use to request **predictions**.
For example, if you use an average value, and for training purposes you average over 10 days, but when you request prediction, you average over the last month.

➔ **Avoid bias**

Make sure that your **training data is representative of the entire universe** of potential data that you will be making predictions for. For example, if you have customers that live all over the world, you should not use training data from only one country.

➔ **Use extra care with imbalanced classes for classification models**

If you have imbalanced classes (a classification problem with one or more outcomes that is seen rarely). Provide sufficient training data for the minority class or undersample the majority class (resampling method).

➔ **Provide enough training data**

If you don't provide enough training data, the resulting model might perform poorly. The more columns you use to train your model, the more data you need to provide.

References

https://www.ibm.com/blogs/watson/2020/11/nlp-vs-nlu-vs-nlg-the-differences-between-three-natural-language-processing-concepts/
https://www.ibm.com/cloud/learn/natural-language-processing
https://www.geeksforgeeks.org/word-embeddings-in-nlp/
https://fr.wikipedia.org/wiki/Word_embedding
https://machinelearningmastery.com/what-are-word-embeddings/
https://machinelearningmastery.com/develop-word-embeddings-python-gensim/
https://medium.com/data-breach/introduction-to-orb-oriented-fast-and-rotated-brief-4220e8ec40cf
https://prince-canuma.medium.com/image-pre-processing-c1aec0be3edf
https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/
https://openclassrooms.com/fr/courses/4470531-classez-et-segmentez-des-donnees-visuelles/5024566-appliquez-vos-premiers-traitements-dimages
https://www.omnisci.com/technical-glossary/feature-engineering
https://en.wikipedia.org/wiki/Feature_engineering
https://towardsdatascience.com/feature-engineering-for-machine-learning-3a5e293a5114

Questions ???