Human-Centered AI
Course

LV 706.046 AK HCI 2020
Mini-Projects Explainable AI

Andreas Holzinger*, Marcus Bloice, Anna Saranti, Deepika Singh,
Bernd Malle, Fleur Jeanquartier, Claire Jean-Quartier

Human-Centered AI Lab (Holzinger Group)

Institute for Interactive Systems and Data Science, TU Graz
Institute for Medical Informatics, Statistics & Documentation, Medical University Graz

*) Explainable AI-Lab, Alberta Machine Intelligence Institute, Edmonton, Canada

*) https://human-centered.ai/seminar-explainable-ai-2019

Course Page: https://human-centered.ai/lv-706-046-ak-hci-2020-explainable-ai

- Check first if you are in the right course:
- LV 706.046, VU, 4,5 ECTS (3 Semester hours)
- 921 Computer Science, 924 Software Engineering
- WK Multimedia Information & Web and Data Science
- 786 Doctoral School Computer Science
- 791 Doctoral School Natural Sciences
- Goal: You work actively on a Mini-project supported by a tutor
- Group size: 1 to max. 3 students
- General topic “explainable AI”

Students Workload, 1 ECTS ≅ 25 to 30 hours *)

*) according to European Commission decision of 12 December 2011
https://ec.europa.eu/education/ects/users-guide/key-features_en.htm

Presence during lecture	8 * 3 h	024 h
Working on the Mini-Projects	85 h	085 h
Presentation in the Mini-Conf including preparation	1 * 3 h	003 h
TOTAL students' workload (minimum**)		112 h

**) On the basis of 1 ECTS = 25 hours

First part: Mini-project and task understanding

Nr	Day, Date	Time	h	Topic
01	Monday 09.03. 2020	18:00	3 h	Introduction and Overview of the course: Presenting the mini projects by the course tutors
02	Monday 23.03. 2020	18:00	3 h	Presenting the Mini-Project topics by the groups at their own for a better mutual understanding

Second part: Work on the mini-projects

03	Monday 30.03. 2020	18:00	3 h	First progress report and discussion of mini-project status (on demand)
04	Monday 20.04. 2020	18:00	3 h	Second progress report and discussion of mini-project status (mandatory)
05	Monday 04.05. 2020	18:00	3 h	Third progress report and discussion of mini-project status (on demand)

Third part: Finalizing the mini-projects

06	Monday 18.05.2020	18:00	3 h	Pre-Presentation of Mini-project results, final feedback round (mandatory)
07	Monday 25.05.2020	18:00	3 h	Feedback-and-Discussion Round (on demand)
08	Monday 08.06.2020	18:00	3h	Pre-Presentation of Mini-Project Results, Final Feedback Round (on demand)

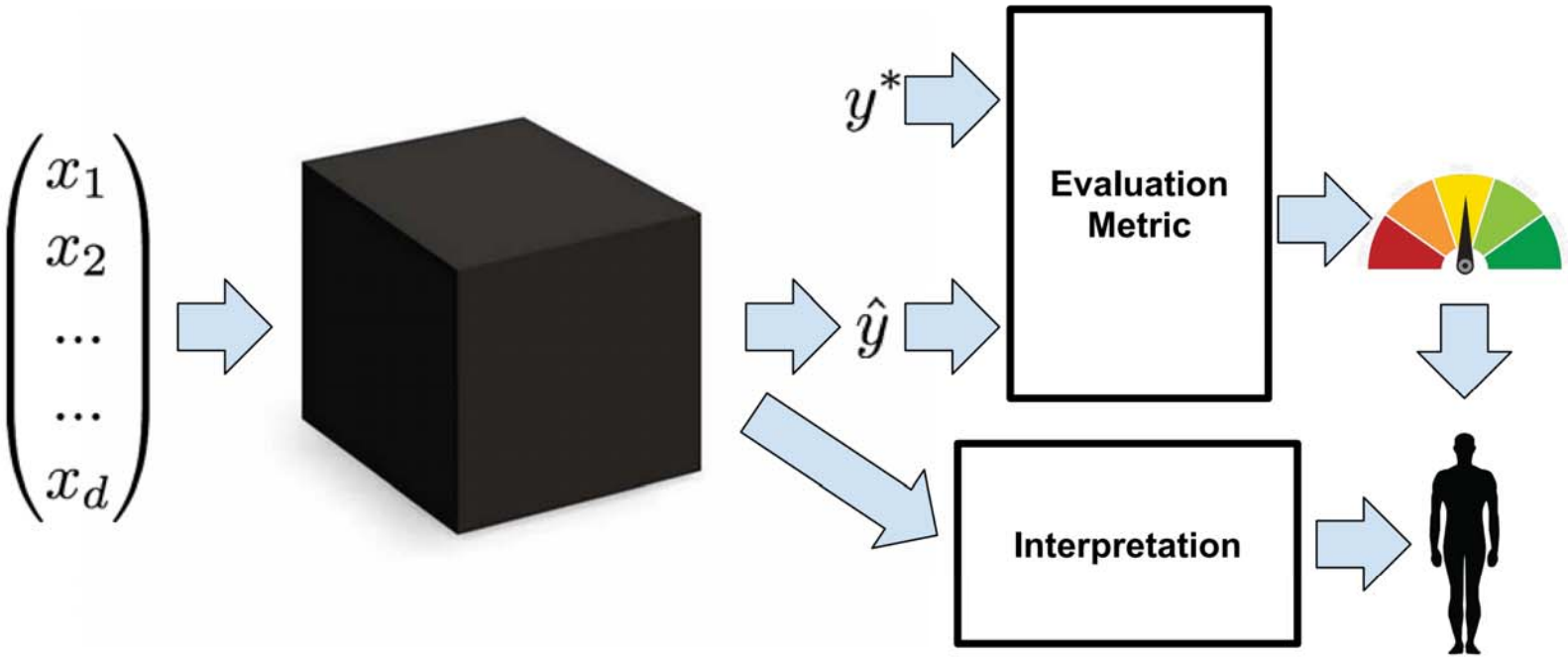
Finale: Presentation of the Mini-projects

09	Monday 22.06.20 20	18:00	3 h	MiniConf (written paper, 60 %) and oral presentation (talk, 40%)

“Paper” = “Technical Report”!

“Explainable AI”

We (now) need interpretable models

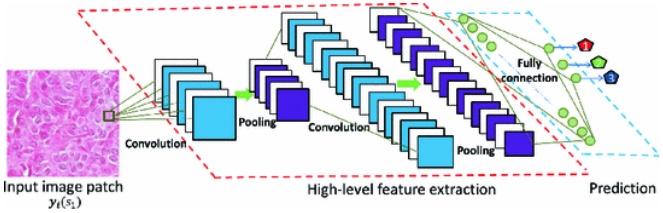


Zachary C. Lipton 2016. The mythos of model interpretability. arXiv:1606.03490.

Why explainable AI?

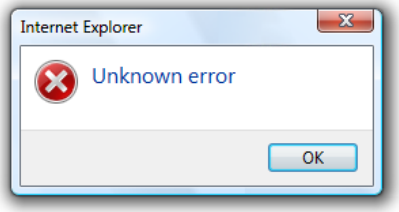
Verify that algorithms/classifiers work as expected …

Wrong decisions can be costly and dangerous …



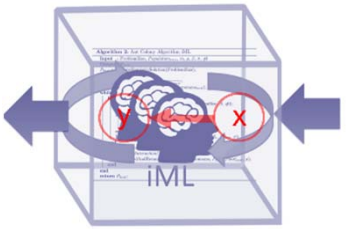
Understanding the errors …

Detection of bias, weaknesses, unknowns, …



Scientific replicability and causality …

The “why” is often more important than the prediction …



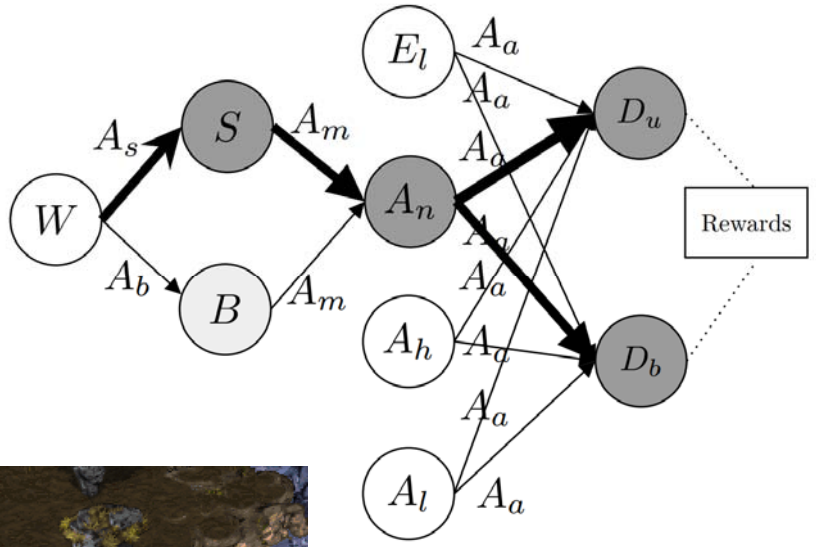
Andreas Holzinger 2018. From Machine Learning to Explainable AI. 2018 World Symposium on Digital Intelligence for Systems and Machines (IEEE DISA). pp. 55-66, doi:10.1109/DISA.2018.8490530.

Differences: Post-hoc versus Ante-hoc

- **Post-Hoc** (latin) = after- this (event), i.e. such approaches provide an explanation for a specific solution of a “black-box” approach, e.g. LIME, BETA, LRP, …
- **Ante-hoc** (latin) = before-this (event), i.e. such methods can be (human) interpreted immanently in the system, i.e. they are transparent by nature (glass box), similar to the "interactive machine Learning" (iML) model.
- Note: Many ante-hoc approaches appear to the new student particularly novel, but these have a long tradition and were used since the early beginning of AI and applied in expert systems, e.g. decision trees, linear regression, random forests, ...

Andreas Holzinger, Chris Biemann, Constantinos S. Pattichis & Douglas B. Kell 2017. What do we need to build explainable AI systems for the medical domain? *arXiv:1712.09923*.

Example: Action Influence Graph of a Starcraft agent



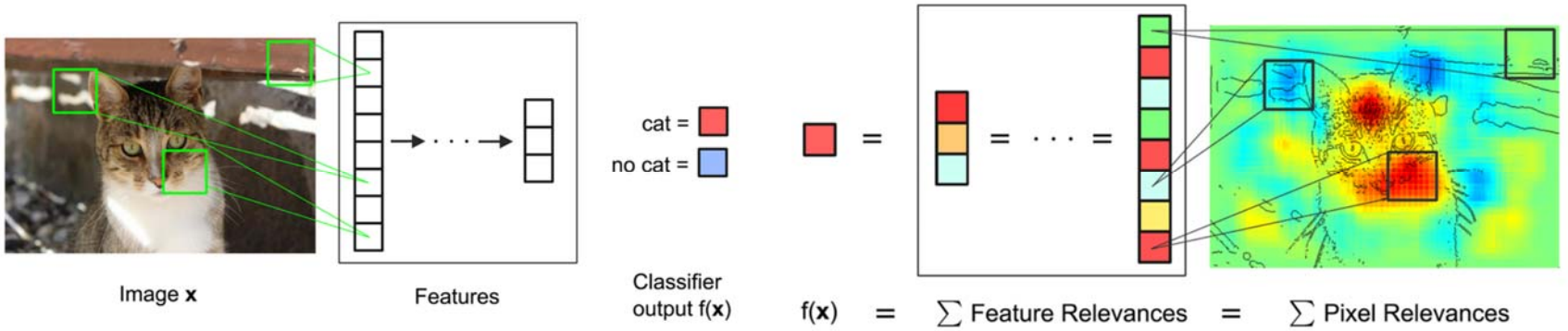
State variables:
W - Worker number
S - Supply depot number
B - barracks number
E - enemay location
A_n - Ally unit number
A_h - Ally unit health
A_l - Ally unit location
D_u - Destoryed units
D_b - Destroyed buildings

Actions:
A_s - build supply depot
A_b - build barracks
A_m - train offensive unit
A_a - attack

https://eecs.wsu.edu/~ala/cdtldms/reports/maf_report.htm

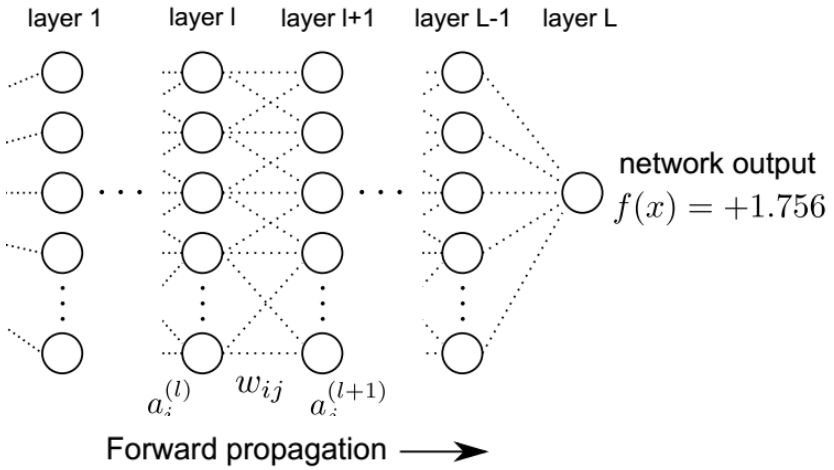
Prashan Madumal, Tim Miller, Liz Sonenberg & Frank Vetere 2019. Explainable Reinforcement Learning Through a Causal Lens. arXiv preprint arXiv:1905.10958.

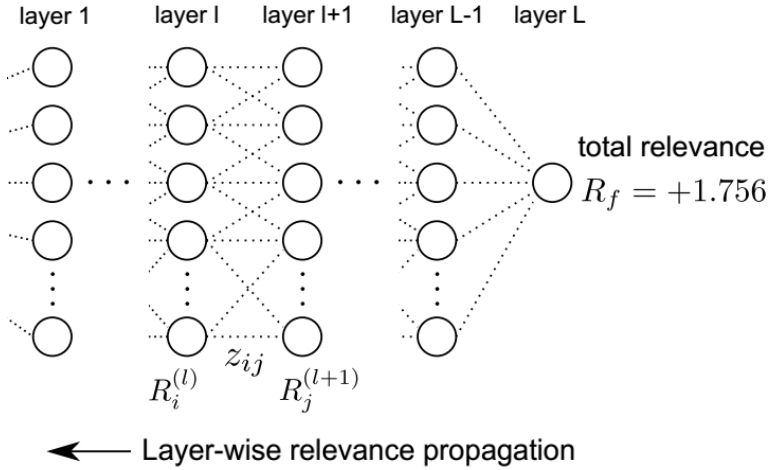
Example: LRP Layer-Wise Relevance Propagation



$$a_j^{(l+1)} = \sigma\left(\sum_i a_i^{(l)} w_{ij} + b_j^{(l+1)}\right)$$

$$R_i^{(l)} = \sum_j \frac{z_{ij}}{\sum_{i'} z_{i'j}} R_j^{(l+1)}$$

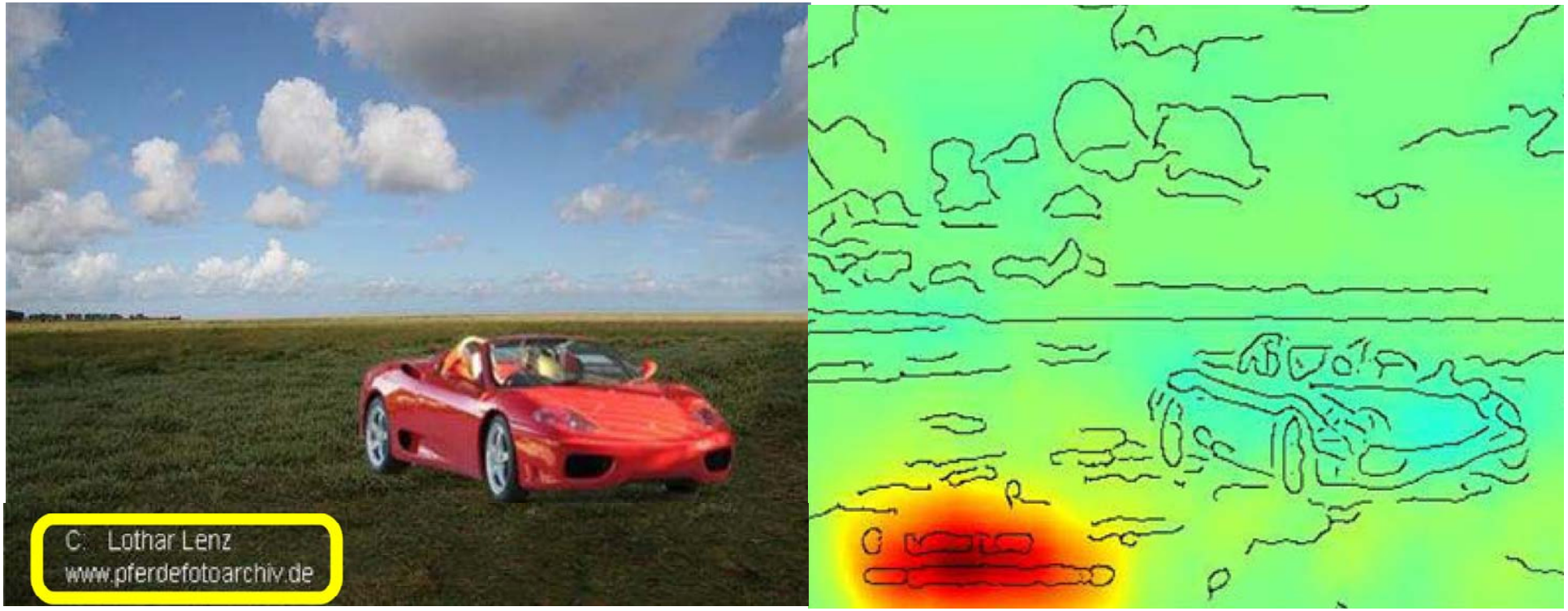




$$R_i = \left|\left|\frac{\partial}{\partial x_i} f(\mathbf{x})\right|\right| \qquad \sum_i R_i = \ldots = \sum_j R_j = \sum_k R_k = \ldots = f(\mathbf{x})$$

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller & Wojciech Samek 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PloS one, 10, (7), e0130140, doi:10.1371/journal.pone.0130140.

98 % A horse on a meadow



Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek & Klaus-Robert Müller 2019. Unmasking Clever Hans predictors and assessing what machines really learn. Nature Communications, 10, (1), doi:10.1038/s41467-019-08987-4.

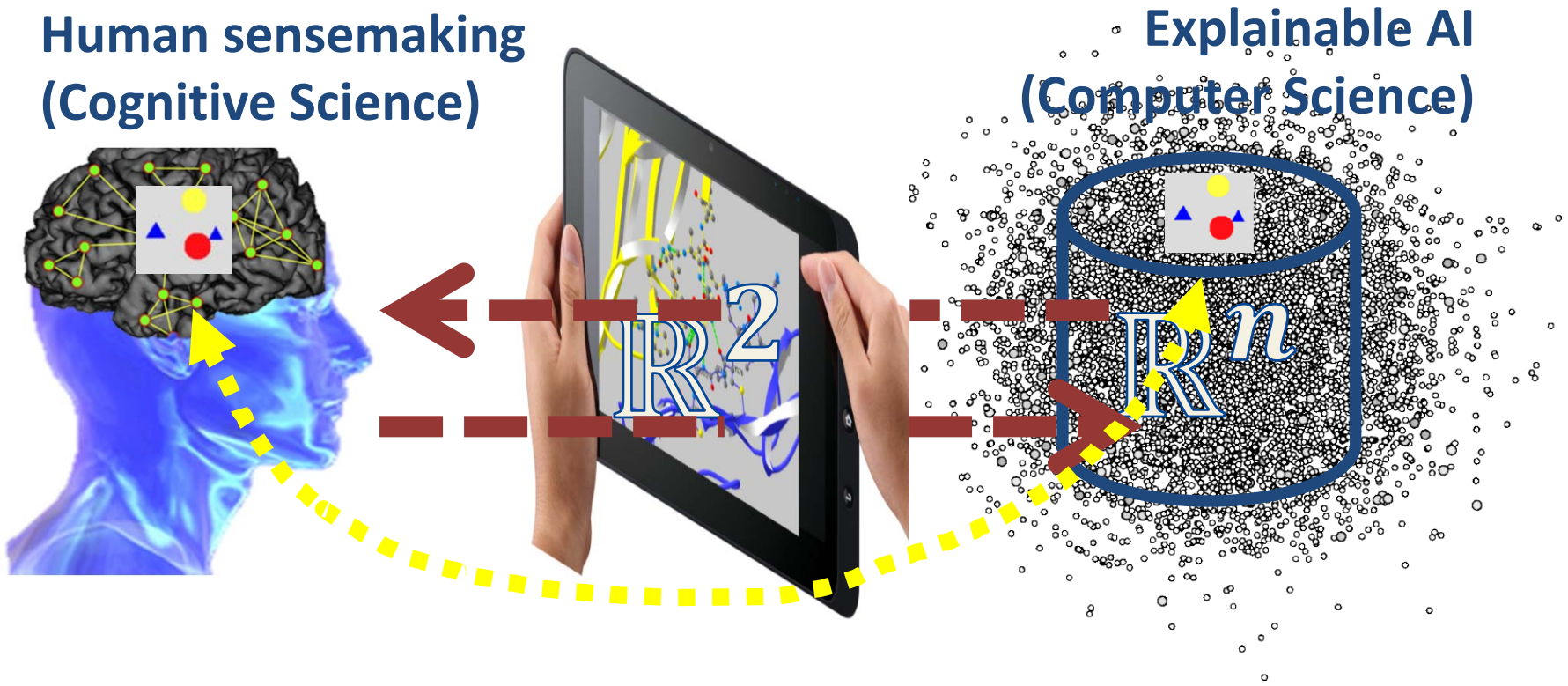
Interpretable Models vs. Interpreting Models

- **Interpretable Models,** the model itself is already interpretable, e.g.
  - Regression
  - Naïve Bayes
  - Random Forests
  - Decision Trees/Graphs
  - …
- **Interpreting Black-Box Models** (the model is not interpretable and needs a post-hoc interpretability method – like a combustion engine ;-) e.g.:
  - Decomposition
  - LIME/BETA
  - LRP
  - (see next slide) …

A final note on Measuring Causability: Mapping machine explanations with human understanding

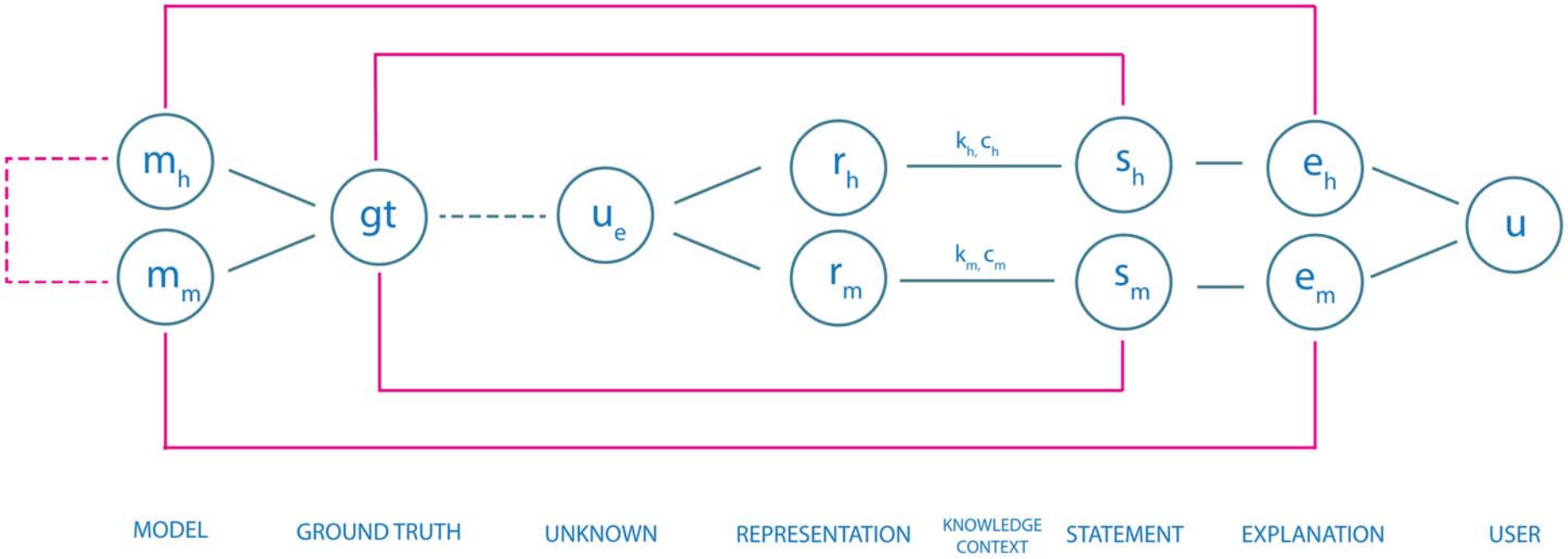
Our goal is to provide effective mapping Human-AI

- Causability := a property of a person (Human)
- Explainability := a property of a system (Computer)



Andreas Holzinger et al. 2019. Causability and Explainability of AI in Medicine. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, doi:10.1002/widm.1312.

Metrics of AI - System Causability Scale



Andreas Holzinger, Andre Carrington & Heimo Müller 2020. Measuring the Quality of Explanations: The System Causability Scale (SCS). Comparing Human and Machine Explanations. KI - Künstliche Intelligenz (German Journal of Artificial intelligence), Special Issue on Interactive Machine Learning, Edited by Kristian Kersting, TU Darmstadt, 34, (2), doi:10.1007/s13218-020-00636-z.

Mini Projects
01 – 17
Explainable AI methods

Methods of ex-AI (I)

- **1) LIME –** Local Interpretable Model Agnostic Explanations

  Marco Tulio Ribeiro, Sameer Singh & Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016. ACM, 1135-1144, doi:10.1145/2939672.2939778.

- **2) BETA –** Black Box Explanations through Transparent Approximation

  Himabindu Lakkaraju, Ece Kamar, Rich Caruana & Jure Leskovec 2017. Interpretable and Explorable Approximations of Black Box Models. arXiv:1707.01154.

- **3) LRP –** Layer-wise Relevance Propagation

  Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller & Wojciech Samek 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PloS one, 10, (7), e0130140, doi:10.1371/journal.pone.0130140.

- **4) DTD -** Deep Taylor Decomposition

  Gregoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek & Klaus-Robert Müller 2017. Explaining nonlinear classification decisions with deep Taylor decomposition. Pattern Recognition, 65, 211-222, doi:10.1016/j.patcog.2016.11.008.

- **5) PDA -** Prediction Difference Analysis

  Luisa M. Zintgraf, Taco S. Cohen, Tameem Adel & Max Welling 2017. Visualizing deep neural network decisions: Prediction difference analysis. arXiv:1702.04595.

https://human-centered.ai/seminar-explainable-ai-2019

6) Visualizing CNN with Deconvolution

Matthew D. Zeiler & Rob Fergus 2014. Visualizing and understanding convolutional networks. In: D., Fleet, T., Pajdla, B., Schiele & T., Tuytelaars (eds.) ECCV, Lecture Notes in Computer Science LNCS 8689. Cham: Springer, pp. 818-833, doi:10.1007/978-3-319-10590-1_53.

7) Inverting CNN

Aravindh Mahendran & Andrea Vedaldi. Understanding deep image representations by inverting them. Proceedings of the IEEE conference on computer vision and pattern recognition, 2015. 5188-5196, doi:10.1109/CVPR.2015.7299155.

8) Guided Backpropagation

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox & Martin Riedmiller 2014. Striving for simplicity: The all convolutional net. arXiv:1412.6806.

9) Deep Generator Networks

Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox & Jeff Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. Advances in Neural Information Processing Systems (NIPS 2016), 2016 Barcelona. 3387-3395.

10) Testing with Concept Activation Vectors

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler & Fernanda Viegas. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors, ICML, 2018. 2673-2682.

- # 11) Sensitivity Analysis

  Karen Simonyan, Andrea Vedaldi & Andrew Zisserman 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv:1312.6034.

- # 12) Gradients: DeepLIFT

  Avanti Shrikumar, Peyton Greenside & Anshul Kundaje 2017. Learning important features through propagating activation differences. arXiv:1704.02685.

- # 13) Gradients: Grad-CAM

  Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh & Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. ICCV, 2017. 618-626.

- # 14) Integrated Gradients

  Mukund Sundararajan, Ankur Taly & Qiqi Yan 2016. Gradients of counterfactuals. arXiv:1611.02639.

- # 15) Recursive NN cell state analysis

  Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song & Rabab Ward 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP), 24, (4), 694-707.

- # 16) Fitted Additive

  Yin Lou, Rich Caruana, Johannes Gehrke & Giles Hooker. Accurate intelligible models with pairwise interactions. Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, 2013. ACM, 623-631.

- # 17) iML with the human-in-the-loop

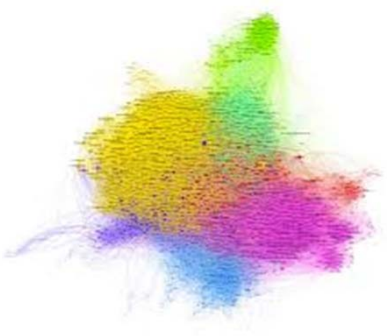
  Andreas Holzinger, Markus Plass, Michael Kickmeier-Rust, Katharina Holzinger, Gloria Cerasela Crişan, Camelia-M. Pintea & Vasile Palade 2019. Interactive machine learning: experimental evidence for the human in the algorithmic loop. Applied Intelligence, 49, (7), 2401-2414, doi:10.1007/s10489-018-1361-5.

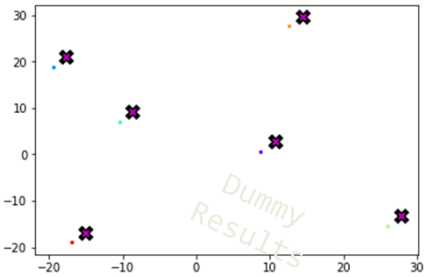
Mini-Projects 01 - 17 Your tasks step-by-step

- 1) Select a model (e.g. NN) of your choice
- 2) Select a dataset of your choice
- 3) Select one method from the methods above
- 4) Document and report your lessons learned, i.e. what insights did you get regarding the selected NN, the selected data and the selected method; what could you change in your NN/dataset/method do get better results in both performance AND explanation
- 5) Write up your report and present your lessons learned during the “Mini-Conf” at the end

Mini Projects
18 – 19
Explainable AI for Cancer Research

Clustering for finding age group specifics in brain tumor diseases





Brief Explanation:

- We will provide genomic data as well as special thematic guidance
- Make use of a clustering algorithm and implement the chosen model with the ability of transparently communicating **WHY** a specific tumor disease clusters related to similarities in age differences occur

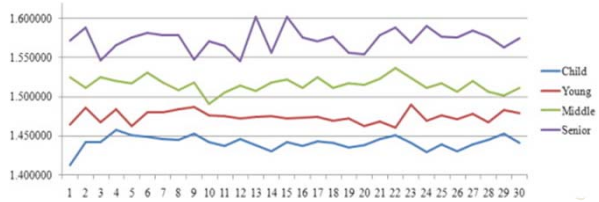
Expected results:

- Tool processing data, and by self-learning providing age based tumor disease clusters + reproducible description

Required knowledge:

- 1. Know-how of ML framework OR 2. Python foundational level and willingness to figure out working with a framework like Pytorch, Tensorflow (Theano)... OR 3. R implementation
- Know-how of processing different data formats

Classification for age based differences in brain tumor diseases



Brief Explanation:

- We will provide genomic data both for training and testing, as well as special thematic guidance (biomedical expertise)
- Make use of a classification algorithm, extract features, identify feature sets and find out **WHAT** are typical feature levels for specific age-based tumor disease classes and communicate the results transparently

Expected results:

- Tool processing data, and by self-learning providing age based tumor disease classes + reproducible description

Required knowledge:

- 1. Know-how of ML framework OR 2. Python foundational level and willingness to figure out working with a framework like Pytorch, Tensorflow (Theano)... OR 3. R implementation
- Know-how of processing different data formats

References

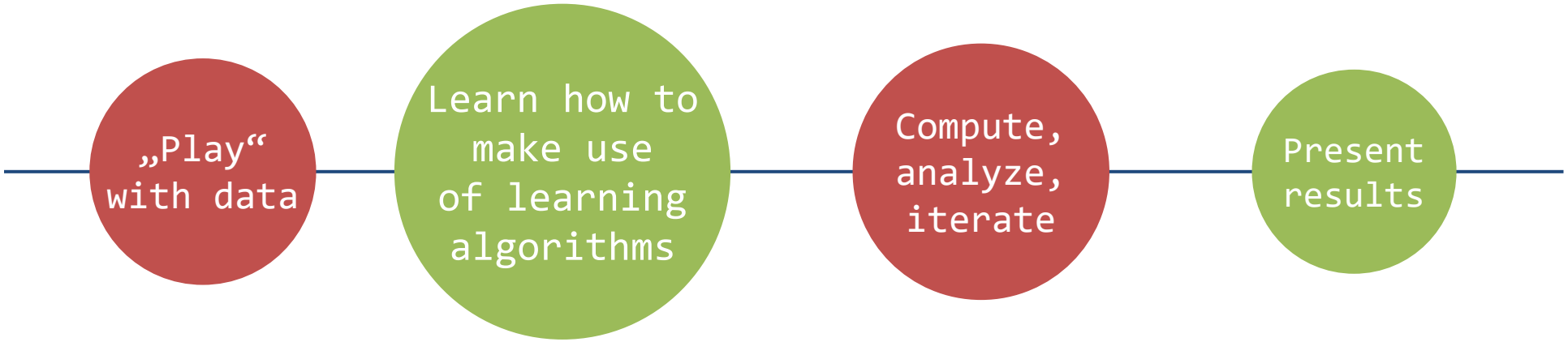
- also consult the course homepage for updates:
- https://human-centered.ai/lv-706-046-ak-hci-2020-explainable-ai
- And the project homepage:
- https://human-centered.ai/project/tumor-growth-machine-learning

Joseph M De Guia, Madhavi Devaraj & Carson K Leung. DeepGx: deep learning using gene expression for cancer classification. Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, 2019. 913-920.

Fahad Alhasoun, Faisal Aleissa, May Alhazzani, Luis G Moyano, Claudio Pinhanez & Marta C González 2018. Age density patterns in patients medical conditions: A clustering approach. PLoS computational biology, 14, (6), e1006115.

Fleur Jeanquartier, Claire Jean-Quartier & Andreas Holzinger 2019. Use case driven evaluation of open databases for pediatric cancer research. BioData Mining, 12, (1), 2, doi:10.1186/s13040-018-0190-8.

Process & Outlook



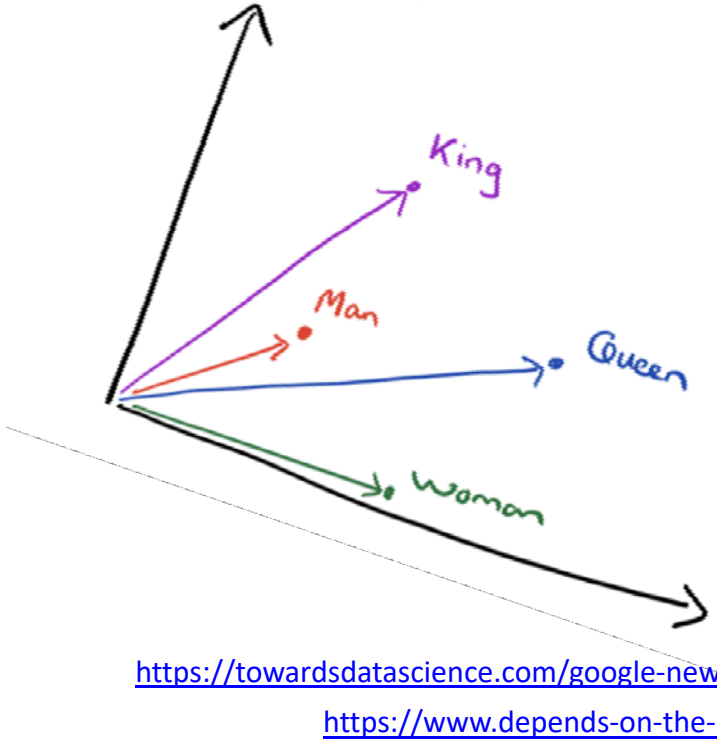
Mini Projects
20 – 21
Explainable AI for Graphical Models

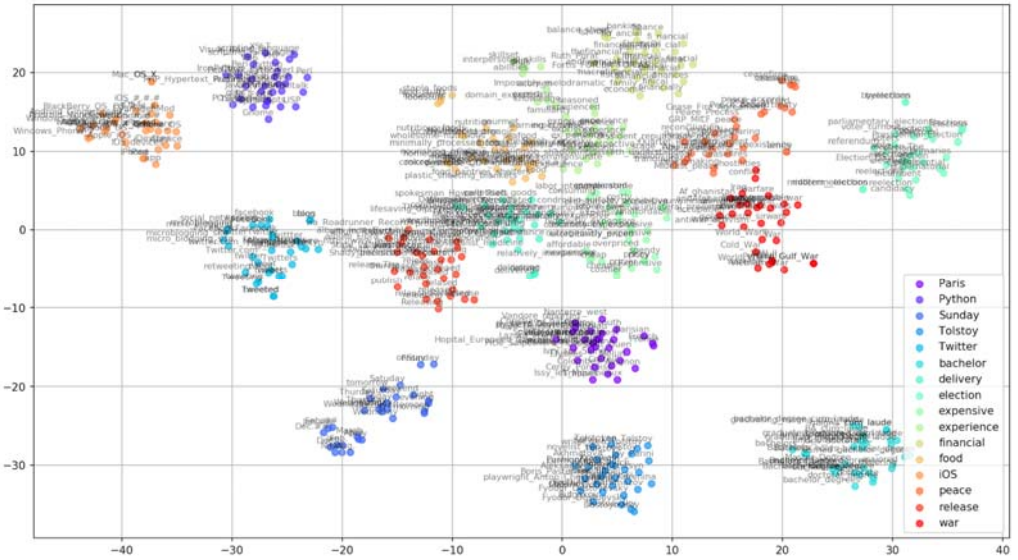
General

- Both mini-projects offer a lot of freedom as far as details are concerned – I will provide some general goals, but you are welcome to go on your own forays!
- Underlying data for both tasks are e-commerce datasets, since those are readily available & unproblematic from a privacy standpoint
- Task details, materials and code will follow within the next 2 weeks – we are still collecting all the parts & assembling the pipelines ;-)

1. Visualization & Analysis of Word Vector Embeddings

- of different models
- via different techniques
- describing (explaining?) how they discriminate / cluster

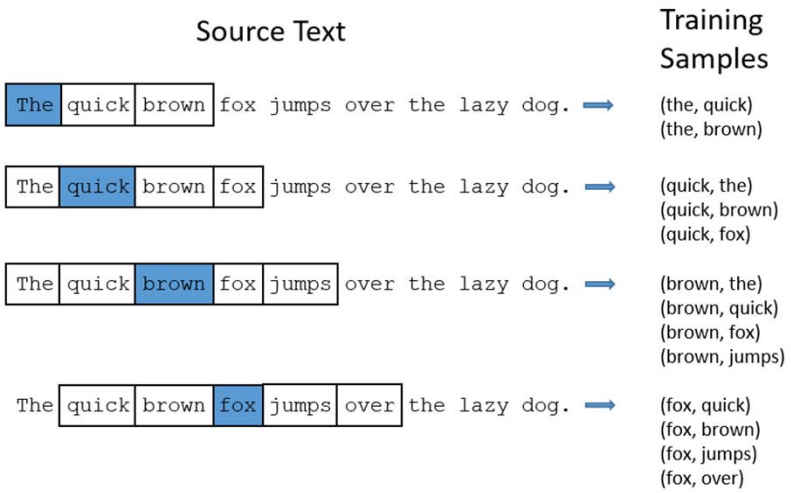


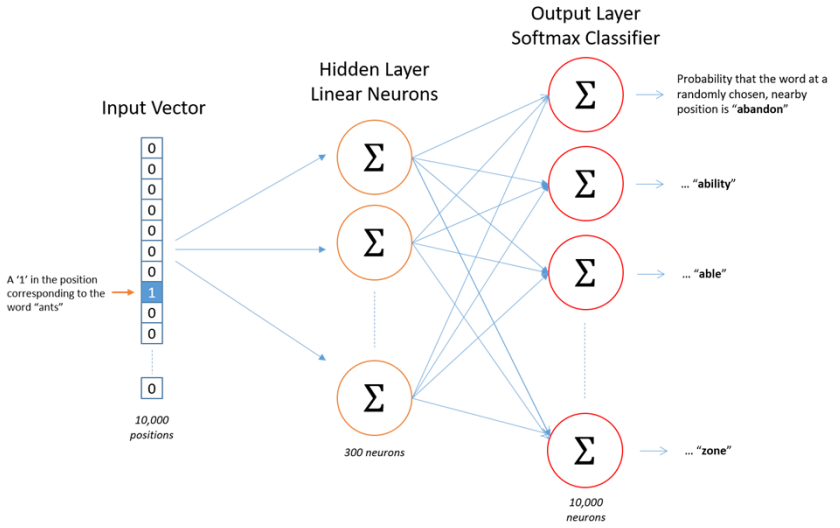


https://towardsdatascience.com/google-news-and-leo-tolstoy-visualizing-word2vec-word-embeddings-with-t-sne-11558d8bd4d

https://www.depends-on-the-definition.com/guide-to-word-vectors-with-gensim-and-keras/

*"**Word embeddings** are mathematical models that encode **word** rel ations within a **vector** space. They are created by an unsupervised training process based on **cooccurrence** information between wordsi n a large corpus"*





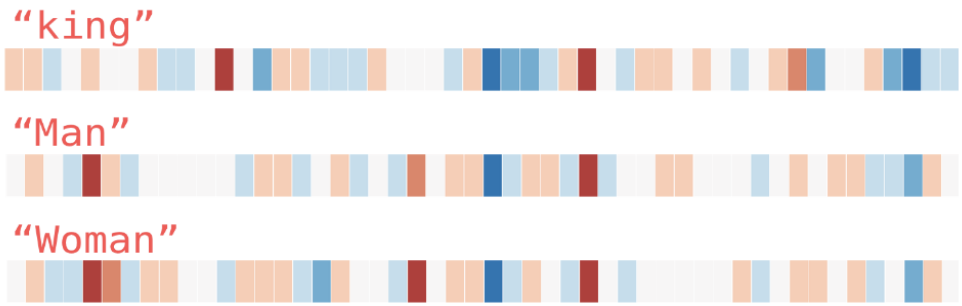
"define a fake task for the NN"
- context prediction in the case of skip-gram

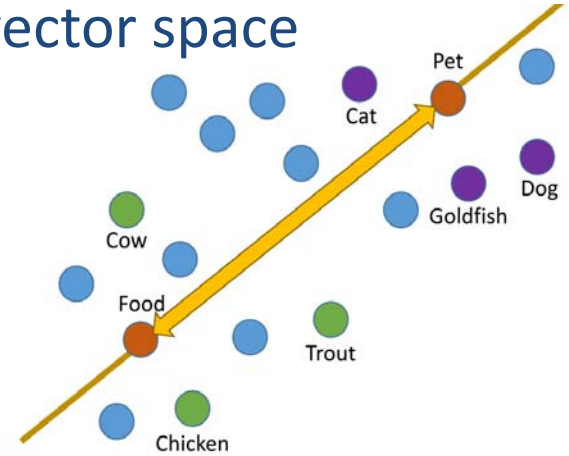
"build a shallow architecture, train & throw away the outputs"
- we only need the embeddings

http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

Tasks:

- learn how to interpret similarity in the vector space



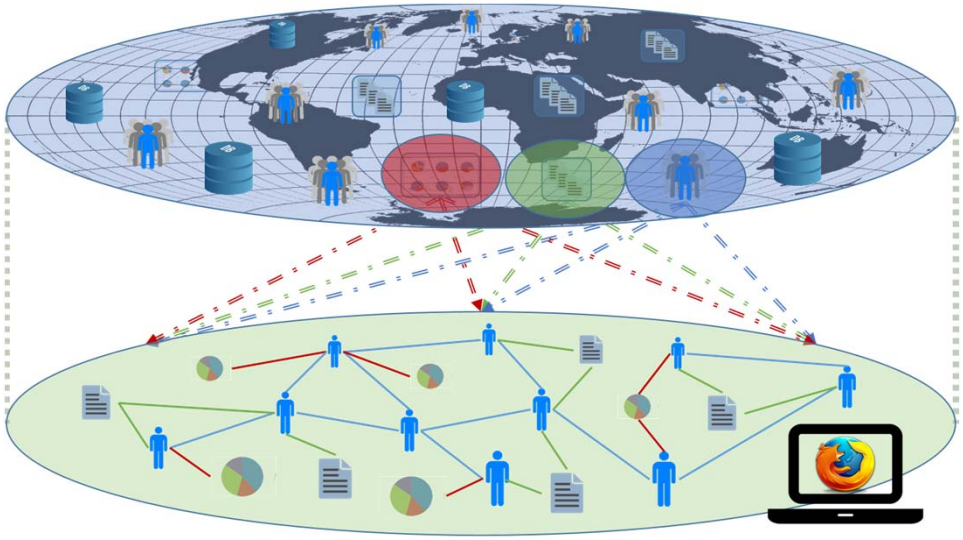


- understand the influence of source corpora on the embeddings
- analyze neighborhoods – what is (un)expected?
- predict & test consequences of changes in parameter settings / pre-processing of input data on the resulting model
- *=> "develop an intuition for embeddings as a basis for future explanations"*

Heimerl, F., & Gleicher, M. (2018). Interactive Analysis of Word Vector Embeddings. *Computer Graphics Forum*, *37*(3). doi:10.1111/cgf.13417

http://jalammar.github.io/illustrated-word2vec/

Visualization of personal recommender graphs (*"Local Sphere"*) & their change over time

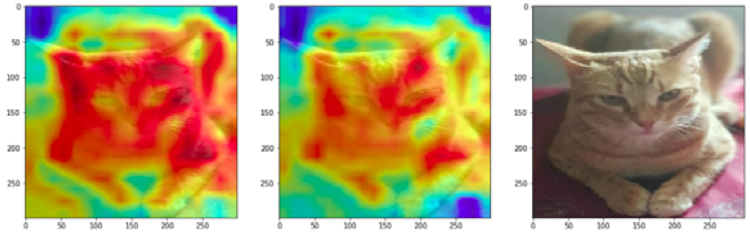


Drawing from globally available resources (e.g. a webshop database en-riched similarities) each user derives her own local sub-graph representing her context / potential interests. As she interacts with the system (explores & follows / ignores visual clues), this context is refined & the graph should respond accordingly.

Bernd Malle, Nicola Giuliani, Peter Kieseberg, and Andreas Holzinger. The More the Merrier - Federated Learning from Local Sphere Recommendations. In Machine Learning and Knowledge Extraction, IFIP CD-MAKE, Lecture Notes in Computer Science LNCS 10410, pages 367–374. Springer, Cham, 2017. doi: 10.1007/978-3-319-66808-6 24.

Project 2 – Background

- Explainability of deep learning systems is a must, but still in the early stages (except for cat pics ;-)



- non-visual and / or higher dimensional data are not intuitive to the human brain – decisions made in those spaces aren't either

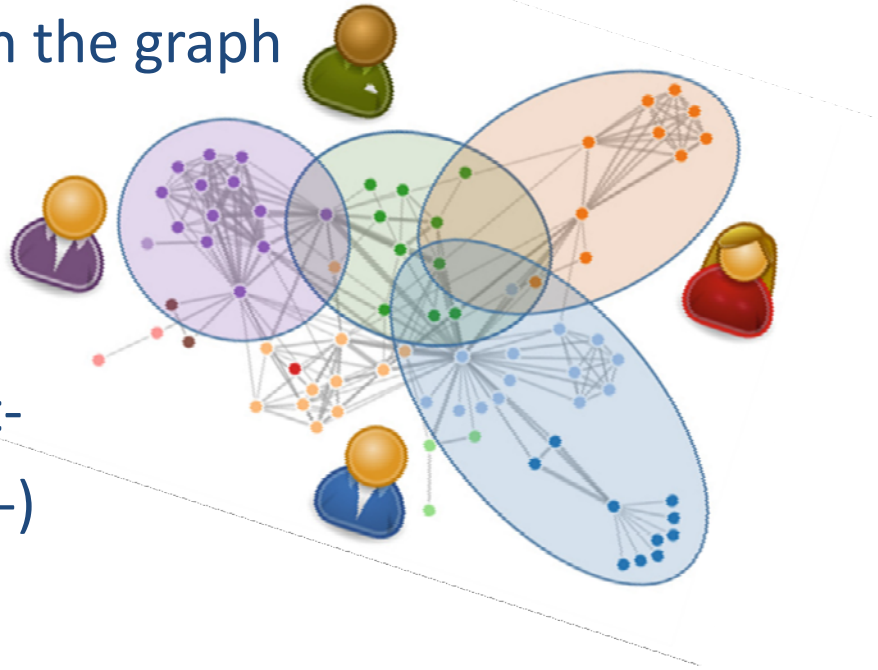
- graphs are a convenient way to break-down high-dimensional information by reducing their complexity to concepts like similarity, connection, and influence.

- understanding which graph metrics will change with user interactions will help develop an intuition about what factors in the original high-dimensional space are relevant for decisions!

https://medium.com/google-developer-experts/interpreting-deep-learning-models-for-computer-vision-f95683e23c1d

Tasks:

- Research graph visualization algorithms pertinent to recommenders (node & edge types, cluster)
- Either extend our existing (Graphinius) VIS library or decide on a different one (but make sure it's properly extensible)
- Highlight recommendations and influence factors (if available)
- Visualize continuous changes in the graph due to user interaction (vids)
- If time permits, visualize several local spheres together
- Graphs, recommender & event-stream will be provided by us ;-)



Mini Projects
22 – 23
Explainable AI &
Text Augmentation

Text Augmentation Project

AK HCI 2020

M.D. Bloice

marcus.bloice@medunigraz.at

9/3/2020

Introduction

This project is about creating a **Python package** for **text augmentation**.

Introduction

This project is about creating a **Python package** for **text augmentation**.

You may be asking:

- What is **text augmentation**?
- What is **augmentation**?

Introduction

This project is about creating a **Python package** for **text augmentation**.

You may be asking:

- What is **text augmentation**?
- What is **augmentation**?

So first, we will discuss a little bit about **augmentation in general**, then **image augmentation**, and finally we will talk about **text augmentation specifically**.

What is Augmentation?

- *Augmentation is the generation of new data through the manipulation of an original data set*

What is Augmentation?

- *Augmentation is the generation of new data through the manipulation of an original data set*
- In other words, you **create data from data**

What is Augmentation?

- *Augmentation is the generation of new data through the manipulation of an original data set*
- In other words, you **create data from data**
- Augmentation is performed often in machine learning and deep learning because you can **get data for free**, and **more data $\approxeq$ more general algorithms**

What is Augmentation?

- *Augmentation is the generation of new data through the manipulation of an original data set*
- In other words, you **create data from data**
- Augmentation is performed often in machine learning and deep learning because you can **get data for free**, and **more data $\approxeq$ more general algorithms**
- It is mostly used for **image data** but also quite a lot for **text data**

What is Augmentation?

- *Augmentation is the generation of new data through the manipulation of an original data set*
- In other words, you **create data from data**
- Augmentation is performed often in machine learning and deep learning because you can **get data for free**, and **more data** $\approxeq$ **more general algorithms**
- It is mostly used for **image data** but also quite a lot for **text data**

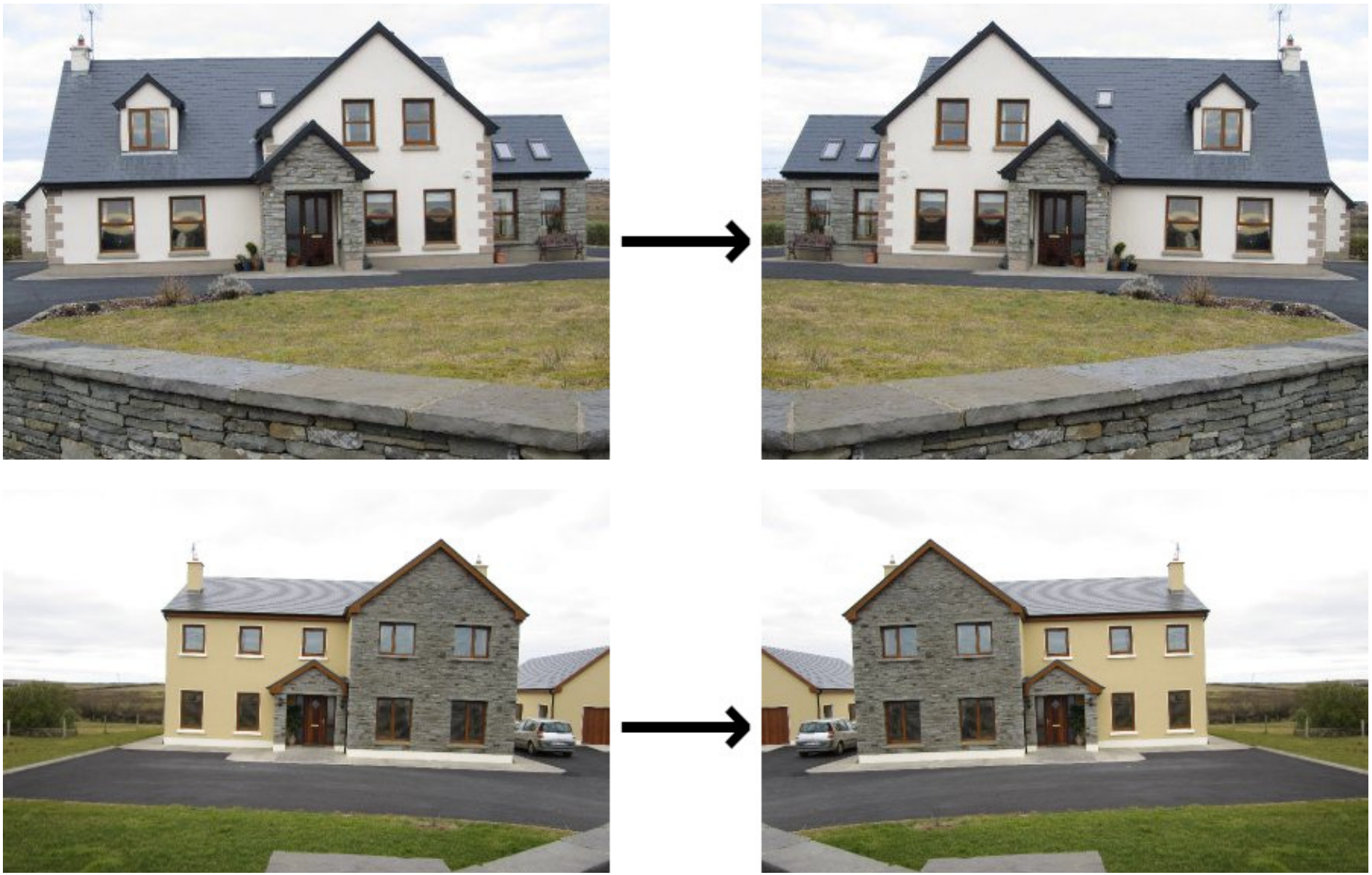
Augmentation is easy to demonstrate with images, so we will do so here...

Image Augmentation Example

Say you have a dataset of houses:

Image Augmentation Example

You can **double** the size of your data set just by flipping them:



Augmentor

Let's take a look at an image augmentation Python package: **Augmentor**

Augmentor

Let's take a look at an image augmentation Python package: **Augmentor**

- As you have seen *Augmentor* is a project for **image augmentation**

Augmentor

Let's take a look at an image augmentation Python package: **Augmentor**

- As you have seen *Augmentor* is a project for **image augmentation**
- Augmentor **started as a project on this course**, and now has nearly 4,000 stars on GitHub!!!

Augmentor

Let's take a look at an image augmentation Python package: **Augmentor**

- As you have seen *Augmentor* is a project for **image augmentation**
- Augmentor **started as a project on this course**, and now has nearly 4,000 stars on GitHub!!!
- **The goal of this project is to create a text version of Augmentor:**
  - **We will call it Augmentext!**

Augmentor

Let's take a look at an image augmentation Python package: **Augmentor**

- As you have seen *Augmentor* is a project for **image augmentation**
- Augmentor **started as a project on this course**, and now has nearly 4,000 stars on GitHub!!!
- **The goal of this project is to create a text version of Augmentor:**
  - **We will call it Augmentext!**
- I have already started the skeleton of the project on GitHub, we will look at this later

Augmentor

Let's take a look at an image augmentation Python package: **Augmentor**

- As you have seen *Augmentor* is a project for **image augmentation**
- Augmentor **started as a project on this course**, and now has nearly 4,000 stars on GitHub!!!
- **The goal of this project is to create a text version of Augmentor:**
  - **We will call it Augmentext!**
- I have already started the skeleton of the project on GitHub, we will look at this later

First let's take a look at how you might structure a text augmentation package...

Augmentation Pipelines

Augmentor uses a **pipeline** based approach to data augmentation.

- What does it mean to have a pipeline-based approach?

Augmentation Pipelines

Augmentor uses a **pipeline** based approach to data augmentation.

- What does it mean to have a pipeline-based approach?
- It means you generate a sequence of **augmentation operations** that are arranged in a sequence

Augmentation Pipelines

Augmentor uses a **pipeline** based approach to data augmentation.

- What does it mean to have a pipeline-based approach?
- It means you generate a sequence of **augmentation operations** that are arranged in a sequence
- Data is passed through this pipeline and the operation is applied, according to a probability, as the data passes through

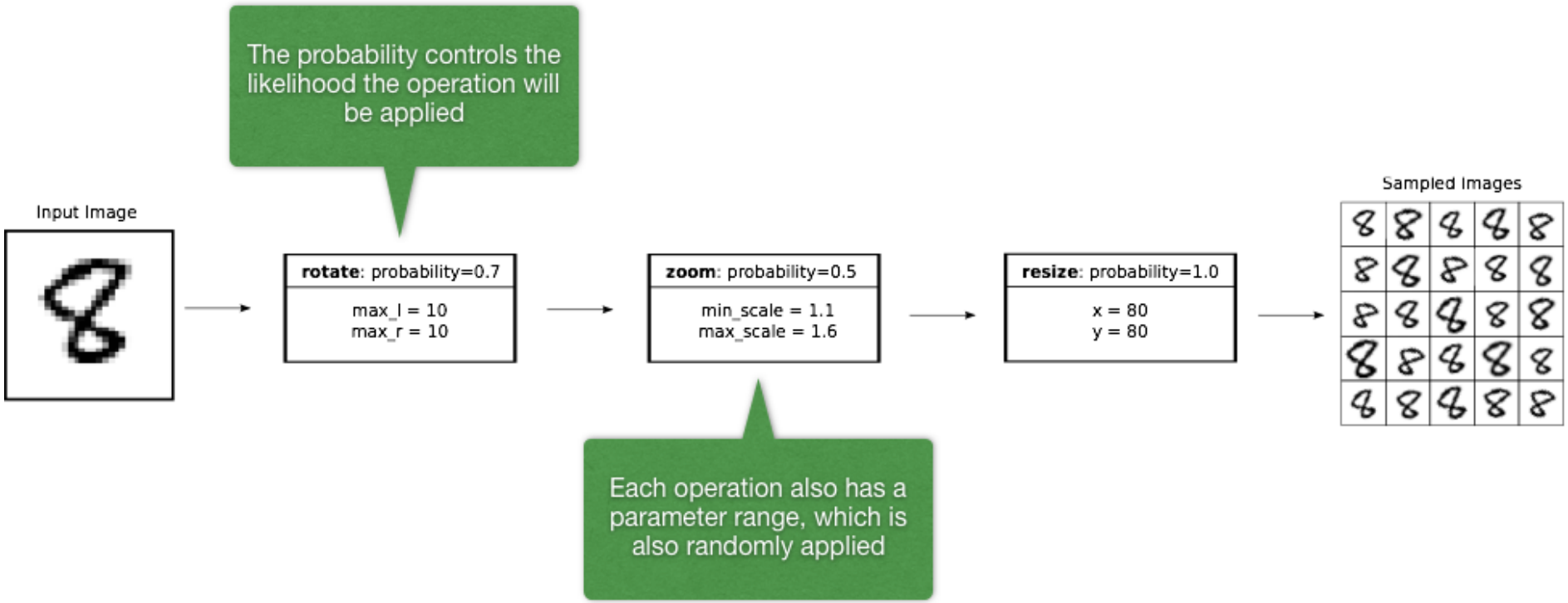
Augmentation Pipelines

Augmentor uses a **pipeline** based approach to data augmentation.

- What does it mean to have a pipeline-based approach?
- It means you generate a sequence of **augmentation operations** that are arranged in a sequence
- Data is passed through this pipeline and the operation is applied, according to a probability, as the data passes through

I would like to follow this pattern for the text augmentation project.

Pipeline Example



Pipeline Example

```
import Augmentor
p = Augmentor.Pipeline("/home/project/data/")

# Rotate randomly between -10 and +10 degrees
p.rotate(probability=0.7, max_left_rotation=10,
            max_right_rotation=10)

# Zoom randomly from between 1.1x and 1.6x
p.zoom(probability=0.5, min_scale=1.1, max_scale=1.6)

# Resize every image by setting probability to 1!
p.resize(probability=1, x=80, y=80)

p.sample(100)
```

Text Augmentation: Examples

So **moving back to text augmentation...**

What are some examples of text augmentation?

Text Augmentation: Examples

So **moving back to text augmentation...**

What are some examples of text augmentation?

- Purposely misspelling words:
  - "the small brown fox" $\rightarrow$ "the smll brown fox"

Text Augmentation: Examples

So **moving back to text augmentation...**

What are some examples of text augmentation?

- Purposely misspelling words:
  - "the small brown fox" $\rightarrow$ "the smll brown fox"
- Swapping word order:
  - "the small brown fox" $\rightarrow$ "the brown small fox"

Text Augmentation: Examples

So **moving back to text augmentation...**

What are some examples of text augmentation?

- Purposely misspelling words:
  - "the small brown fox" $\rightarrow$ "the smll brown fox"
- Swapping word order:
  - "the small brown fox" $\rightarrow$ "the brown small fox"
- Replacing digits with text:
  - "2 small brown foxes" $\rightarrow$ "two small brown foxes"

Text Augmentation: Examples

So **moving back to text augmentation...**

What are some examples of text augmentation?

- Purposely misspelling words:
  - "the small brown fox" $\rightarrow$ "the smll brown fox"
- Swapping word order:
  - "the small brown fox" $\rightarrow$ "the brown small fox"
- Replacing digits with text:
  - "2 small brown foxes" $\rightarrow$ "two small brown foxes"
- Dictionary-based thesaurus word replacement:
  - "the small brown fox" $\rightarrow$ "the slight brown fox"

Text Augmentation: Differences

Text augmentation is fundamentally different to image augmentation in several ways.

For example, the way in which text can be structured...

[1] Source: https://github.com/oxinabox/MultiResolutionIterators.jl

Text Augmentation: Differences

Text augmentation is fundamentally different to image augmentation in several ways.

For example, the way in which text can be structured...

In text, we have this idea of a **corpus**[1]:

- Corpus:
  - made up of: Documents
  - made up of: Paragraphs
  - made up of: Sentences
  - made up of: Words
  - made up of: Characters

[1] Source: https://github.com/oxinabox/MultiResolutionIterators.jl

Text Augmentation: Differences

Text augmentation is fundamentally different to image augmentation in several ways.

For example, the way in which text can be structured...

In text, we have this idea of a **corpus**[1]:

- Corpus:
  - made up of: Documents
  - made up of: Paragraphs
  - made up of: Sentences
  - made up of: Words
  - made up of: Characters

How should we handle this type of data? I think we could do so with a pipeline-based approach.

[1] Source: https://github.com/oxinabox/MultiResolutionIterators.jl

Text Augmentation: Pipelines

As I have mentioned (several times now ☺), I think a good approach would be to mirror the Augmentor project and to use pipelines. Something like:

Text Augmentation: Pipelines

As I have mentioned (several times now ☺), I think a good approach would be to mirror the Augmentor project and to use pipelines. Something like:

```
import Augmentext
p = Augmentext.Pipeline(text_dataframe)

# Misspell every 100th word
p.misspell(probability=0.01)

# Swap every 100th word with a random neighbour
p.swapwords(probability=0.01)

# Generate 100 sentences of text
p.sample(100)
```

Text Augmentation: Pipelines

As I have mentioned (several times now ☺), I think a good approach would be to mirror the Augmentor project and to use pipelines. Something like:

```
import Augmentext
p = Augmentext.Pipeline(text_dataframe)

# Misspell every 100th word
p.misspell(probability=0.01)

# Swap every 100th word with a random neighbour
p.swapwords(probability=0.01)

# Generate 100 sentences of text
p.sample(100)
```

So, part of this project would be to work out the best way to handle text data and how text should be internally represented in the Augmentext package!

NLP Libraries

Seems daunting? Luckily, because we will use Python, there are many libraries that can help:

NLP Libraries

Seems daunting? Luckily, because we will use Python, there are many libraries that can help:

- **TextBlob**: TextBlob is a Python library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

NLP Libraries

Seems daunting? Luckily, because we will use Python, there are many libraries that can help:

- **TextBlob**: TextBlob is a Python library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.
- **NLTK**: Natural Language Toolkit: platform for building Python programs to work with human language data. It interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, and wrappers for industrial-strength NLP libraries.

NLP Libraries

Seems daunting? Luckily, because we will use Python, there are many libraries that can help:

- **TextBlob**: TextBlob is a Python library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.
- **NLTK**: Natural Language Toolkit: platform for building Python programs to work with human language data. It interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, and wrappers for industrial-strength NLP libraries.
- **spaCy**, **Gensim**, **Pattern**, etc. etc.

Augmentext: Where to start...

Augmentext: Where to start...

- As you can see we already have a logo ☺

Augmentext: Where to start...

- As you can see we already have a logo ☺
- We know that several libraries exist that can be used to parse text

Augmentext: Where to start...

- As you can see we already have a logo ☺
- We know that several libraries exist that can be used to parse text
- We know we want to use a **pipeline-based approach but for text**

Augmentext: Where to start...

- As you can see we already have a logo ☺
- We know that several libraries exist that can be used to parse text
- We know we want to use a **pipeline-based approach but for text**
- So, I have already created **some skeleton code with the basic logic for a pipeline**

Augmentext: Where to start...

- As you can see we already have a logo ☺
- We know that several libraries exist that can be used to parse text
- We know we want to use a **pipeline-based approach but for text**
- So, I have already created **some skeleton code with the basic logic for a pipeline**

The skeleton code already includes some logic for the pipeline:

https://github.com/mdbloice/Augmentext

Therefore, you are not starting at zero!

Currently the project is marked as **private**, until we get to a state which is at least v0.1 beta or so ☺

Project Prerequisites

What are the prerequisites for this project? These might be useful:

- Familiarity with Python
- Would be nice to have experience with Git & GitHub
- Some NLP (Natural Language Processing) experience would be really great, but I do not think essential
- **By no means do I expect a finished package at the end of this semester!!!**

Project Prerequisites

What are the prerequisites for this project? These might be useful:

- Familiarity with Python
- Would be nice to have experience with Git & GitHub
- Some NLP (Natural Language Processing) experience would be really great, but I do not think essential
- **By no means do I expect a finished package at the end of this semester!!!**

The package will be:

- Open-source, free, MIT licensed
- Hosted on GitHub

All that good stuff... this means you must be willing to give your work away for free ☺

Tasks for Next Week

So **if you do the project** your tasks for next week would be:

1. **What kind of data structure should we use for augmenting textual data?**

Tasks for Next Week

So **if you do the project** your tasks for next week would be:

1. **What kind of data structure should we use for augmenting textual data?**
2. Check out the other text augmentation packages that exist and see **how they work**:
   - Do they use pipelines?
   - What kind of operations do they support?

Tasks for Next Week

So **if you do the project** your tasks for next week would be:

1. **What kind of data structure should we use for augmenting textual data?**
2. Check out the other text augmentation packages that exist and see **how they work**:
   - Do they use pipelines?
   - What kind of operations do they support?
3. Think of a few **augmentation operations** (we have seen several already)

Tasks for Next Week

So **if you do the project** your tasks for next week would be:

1. **What kind of data structure should we use for augmenting textual data?**
2. Check out the other text augmentation packages that exist and see **how they work**:
   - Do they use pipelines?
   - What kind of operations do they support?
3. Think of a few **augmentation operations** (we have seen several already)

Organisation:

- Contact: marcus.bloice@medunigraz.at
- Slack: https://augmentext.slack.com
- GitHub: https://github.com/mdbloice/Augmentext

Any questions? Ask now!