TUTORIAL

INTELLIGIBLE MACHINE LEARNING AND KNOWLEDGE DISCOVERY BOOSTED BY VISUAL MEANS
TUTORIAL:
INTELLIGIBLE MACHINE LEARNING AND KNOWLEDGE DISCOVERY BOOSTED BY VISUAL MEANS

Prof. Boris Kovalerchuk
Dept. of Computer Science,
Central Washington University, USA

https://dl.acm.org/doi/abs/10.1145/3336191.3371872

CWU Tutorial Website
http://www.cwu.edu/~borisk/WSDM
Schedule

- February 3, 2020
- Time: Half-Day
  - 2:00 pm start
  - 3:00-3:30 pm Break
- Room: Rice
- Hyatt Regency Houston/Galleria
Agenda and sources

■ Motivation
■ Recent eXplainable AI (XAI) methodologies with visual means:
  – How to understand instance explanation and model explanation using visualization (explain and discover what ML models are telling us with visual means)?
  – How to discover explainable analytical ML models using visual means?
  – How to discover explainable visual ML models using analytical ML algorithms?
  – How to visualize very large datasets, embeddings, deep models, etc?
■ Case Studies

■ Sources: Multiple relevant publications including presenter’s books:
  – “Visual and Spatial Analysis: Advances in Visual Data Mining, Reasoning, and Problem Solving” (Springer), and
Overview

- This tutorial covers the state-of-the-art research, development, and applications of
  - *interpretable knowledge discovery reinforced by visual methods* to stimulate and facilitate future work.

- The topic is interdisciplinary bridging of scientific research and applied communities in
  - *Machine Learning, Data Mining, KDD, Visual Analytics, Information Visualization, and HCI*.

- This is a novel and fast growing area with significant applications, and potential.

- First these studies have grown under the name of *visual data mining*.

- The recent growth under the names of *deep visualization*, and *visual knowledge discovery*, is motivated considerably by
  - *deep learning success in accuracy of prediction and*
  - *its failure in providing explanation/understanding of the produced models* without special interpretation efforts.

- In the areas of Visual Analytics, Information Visualization, and HCI, the increasing *trend toward machine learning* tasks, including deep learning, is also apparent. providing insight on data mining from a human interface perspective.

- This tutorial reviews progress in these areas with analysis of what each area brings to the joint table.

*The noblest pleasure is the joy of understanding.*

*Leonardo da Vinci*
Dr. Boris Kovalerchuk

- Dr. Boris Kovalerchuk is a professor of Computer Science at Central Washington University, USA. His publications include three books "Data Mining in Finance " (Springer, 2000), "Visual and Spatial Analysis" (Springer, 2005), and "Visual Knowledge Discovery and Machine Learning" (Springer, 2018), chapters in the Data Mining Handbooks (Springer, 2005-2020) and over 170 other publications.

- His research interests are in data mining, machine learning, visual analytics, foundation of uncertainty modeling, data fusion, image and signal processing.

- Dr. Kovalerchuk has been a principal investigator of research projects in these areas supported by the US Government agencies. He served as a senior visiting scientist at the US Air Force Research Laboratory and as a member of expert panels at the international conferences and panels organized by the US Government bodies.

- http://www.cwu.edu/~borisk
Black box vs. White box

Black Box Models

- Deep Learning Neural Networks
- Boosted trees
- Random forests
- ....

Often less interpretable but more accurate.

Glass Box Models

- Single Decision Trees
- Naive-Bayes
- Bayesian Networks
- Propositional and First Order Logic rules

Often less accurate but more interpretable and human understandable.

Often we are forced to choose between accuracy and interpretability. It is a major obstacle to the wider adoption of Machine Learning in areas with high cost of error, such as in cancer diagnosis, and many other domains where it is necessary to understand, validate, and trust decisions. Visual Knowledge discovery helps to get both model accuracy and its explanation.

What is Interpretable Knowledge Discovery, AI/ML?

- Microsoft and Google started eXplainable AI (XAI) services, but do we have an operational definitions of model comprehensibility, interpretability, intelligibility, explainability, understandability?
  - Is a bounding box around the face with facial landmarks provided by Google service an operational explanation without telling in understandable terms how it was derived?

- Can such machines not only learn new concepts, but explain them to humans thereby improving human task performance [Muggleton et al. 2018]?

- Ideas for operational definitions of explanation.

- Donald Michie’s [1998] Machine Learning criteria:
  - Weak criterion – ML improves predictive accuracy with more data.
  - Strong criterion – ML additionally provides its hypotheses in symbolic form.
  - Ultra-strong criterion (comprehensibility)– ML additionally teaches the hypothesis to a human, who consequently performs better than the human studying the training data alone.

- Better comprehension [Gaines, 1996] via improved knowledge representation that is textually smaller, more coherent, composed of familiar concepts.

Explanation approaches

- **Argument-Based ML** (ABML) [Mozina et al. 2007]
  - rule-learning with positive or negative arguments included in some training examples by domain experts as explanations.

- **Explanation-Based Learning** (EBL) [Mitchell et al. 1986], **Meta-Interpretive Learning** (MIL) [Muggleton et al., 2015], **Domain-Measurement Learning** (DML) [Kovalerchuk, Vityaev, 2000].
  - explains how each training example is an instance of the target concept by inferring from background knowledge (domain theory) by first order logic (FOL).

- **Neural Networks Learning.**
  - a system is interpretable if a user can understand how inputs are mathematically mapped to outputs [Doran et al. 2017], e.g., regression and generalized additive models.
  - The system extracts some knowledge from the ML model with subsequent interpretation by domain experts.
  - Requires an active interpretation or technical understanding of this mapping of inputs to outputs.

- All above fail Michie’s ultra-strong test – no user comprehensibility of learned models is ensured.

- **Experimental Studies**
  - Decision tree models are perceived as more understandable than rule-based models.
  - Complexity and understandability are negatively correlated.
  - Users performs better in accuracy, response time, and answer confidence on decision tables than on other representations.

---

Definitions

- **Definition (Comprehensibility, C(S, P))**
  - The comprehensibility of a definition (or program) $P$ with respect to a human population $S$ is the mean accuracy with which a human from population $S$ after brief study and without further sight can use $P$ to classify new material sampled randomly from the definition’s domain.

- **Definition (Inspection time T(S, P))**
  - The inspection time $T$ of a definition (or program) $P$ with respect to a human population $S$ is the mean time a human from $S$ spends studying $P$ before applying $P$ to new material.

- **Definition (Textual complexity, Sz(P))**
  - The textual complexity $Sz$ of a definition of definite program $P$ is the sum of the occurrences of predicate symbols, functions symbols and variables found in $P$.

These definitions allow studying comprehensibility experimentally.

Muggleton et al, 2018
Empirical observations
- Convolutional networks for image classification trained with stochastic gradient methods easily fit a random labeling of the training data.
- It occurs even after replacing the true images by completely unstructured random noise.
  - Here the learning must be impossible and should show up during training, e.g., by not converging or slowing down.
  - Surprisingly, for multiple standard architectures it did not happen.

Theoretical results
- Large neural networks can express any labeling of the training data.
- Theorem. There exists a two-layer neural network with 2n+d weights that can represent any function on a sample of size $n$ in $d$ dimensions.
- These models are in principle rich enough to memorize the training data.
- This theorem is for a finite sample of size $n$ in contrast the NN universal approximation theorems are for the entire domain.

Explanation for such accurate models by known heatmap activation methods can be constructed, but what will be its value?
- To distinguish it from a meaningful explanation we need to analyze the generalization process and errors beyond training data.

How to distinguish between the models trained on the true labels that are potentially explainable and models trained on random labels (high generalization error) that should not be meaningfully explainable?

Human Visual and Oral Generalization vs. Machine Learning Generalization

Examples of Setosa, Versicolor and Virginica Iris petals.

Setosa – small length and small width of petal
Versicolor – medium length and medium width of petal
Virginica - large length and medium to large width of petal

Example of logistic regression classification of these classes of Iris [Big data, 2018].


GLC-L
Classes 2 and 3
Errors 3

Setosa
Versicolor
Virginica

Iris data learning
Human generalization: Experiment

Setting of experiment.

- Participants assigned confidence values from 0 to 10 to ovals 1-4 to class 1 and ovals 5-8 to class 2.
- This experiment shows decreasing confidence from inner ovals to outer ovals for both classes:
  - Inner ovals: confidence 9.73 for class 1 and 9.78 for class 2
  - Outer ovals: confidence 5.91 for class 1 and 6.22 for class 2.

- The experiment shows that participants can control overgeneralization of data in visualization.
- All cases that are far away from training data are classified with low confidence. In essence, people refused to classify them.
- This is in a sharp contrast with automatic ML models where no case was refused.

Example of logistic regression classification of these classes of Iris [Big data, 2018].

Human Visual and Verbal Generalization vs. Machine Learning generalization: Iris data

- Wrong ML prediction (overgeneralization) in this example is not a result of insufficient training data.

- It is a result of the task formulation – search for the linear or non-linear discrimination lines in LDA, SVM and DT that classify every point in the space to one of these three classes.

- Instead we can use envelopes around training data of each class – small length and width of petal for setosa. Points outside of the envelopes are not recognized at all. The algorithm refuses to classify them.

- How can we know that we need to use an envelope not such functions for n-D data?

- We need visualization tools to visualize n-D data in 2-D without loss of n-D information (lossless visualization) allowing to see n-D data as we see 2-D data.

- Visualization will allow to see an n-D granule that corresponds to small petal in linguistic terms expressed in fuzzy sets or subjective probabilities to formalize these linguistic terms.

- Synergy with computing with images [Kovalerchuk, 2013].

Kovalerchuk, B., Quest for rigorous intelligent tutoring systems under uncertainty: Computing with Words and Images, In: Joint IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS), 2013. pp. 685 - 690, DOI: 10.1109/IFSA-NAFIPS.2013.6608483
Human preference experiment: Linear vs. Non-linear discrimination

In this experiment, humans prefer the non-linear solution (7.14 vs. 5.0 with max 10). In contrast many analytical ML methods that search for the simplest solution will prefer a straight line as a more robust solution.

<table>
<thead>
<tr>
<th>Model</th>
<th>mean</th>
<th>stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear</td>
<td>5.0</td>
<td>2.2</td>
</tr>
<tr>
<td>non-linear</td>
<td>7.14</td>
<td>2.17</td>
</tr>
</tbody>
</table>

Highlights

• Goal: machine learning and visualization for discovering hidden understandable patterns in multidimensional data (n-D data).
  – The fundamental challenge -- we cannot see multidimensional data with a naked eye and need visual analytics tools (“n-D glasses”). It starts at 4-D.

• Often we use non-reversible, lossy dimension reduction methods such as Principal Component Analysis that convert, say, every 10-D point (10 numbers) to 2-D point (two numbers) in visualization.
  – They are very useful, but they can remove important multidimensional information before starting discovering complex n-D patterns.

• The hybrid methods combine advantages of reversible and non-reversible methods for interpretable knowledge discovery in n-D data.
APPROACHES
Approaches to visualize Machine Learning (ML) models produced by the analytical ML methods

- These visualization of ML models are used
  - to *demonstrate and understand ML models* such as CNN, Association Rules and others,
  - to show the *superposition* of input data/images with heat maps of model layers,
  - *dataflow* graph in Deep Learning models,
  - *differences in the internal representations* of objects, adversarial cases and others using t-SNE and other visualization methods.
Demonstration of DNN models to understand them via heatmaps visualizing discovered features

ConvNets: Multiple Trainable Layers, Hierarchical Representations

Traditional Pattern Recognition: Fixed/Handcrafted Feature Extractor

Deep Learning: Representations are hierarchical and trained

Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]
Visualization of Association Rules

- Example: rule \{onions, potatoes\} ⇒ \{burger\}
  - If customers buy both onions and potatoes, they are more likely to buy a burger.
- Association rules – interpretable
- Challenges –
  - limited specifics (e.g., why rule confidence is less than 100%?)
  - incomplete understandings of relation strength
  - How to identify uncertainty of patterns, outliers.
  - Deleting many non-interesting rules.
- Quality Measures:
  - Support -- how frequently an itemset appears in the dataset.
  - Confidence, \(\text{conf} (X ⇒ Y) = \frac{\text{supp}(X \& Y)}{\text{supp}(X)}\) -- fraction of transactions with X and Y.
  - Lift, \(\text{lift}(X ⇒ Y) = \frac{\text{supp}(X \& Y)}{(\text{supp}(X) \times \text{supp}(Y))}\) -- the ratio of the observed support to that expected if X and Y are independent.

- Visualization approaches:
  - Matrix with rows as Left Hand Side, LHS itemsets of rules and columns as Right Hand side (RHS) itemsets of rules.
  - Parallel sets

- Challenges:
  - Scalability for many LHS and RHS
  - Readability of small cells having many categories in variables.

Visualization of Association Rules (AR) using Parallel sets for categorical data

- **Approach:**
  - Discovering Association Rules.
  - Deleting dimensions irrelevant to AR.
  - Feeding rules to two coordinated rule visualizations (called ARTable and ParSets),

- **User interactions:**
  - Visually explore rules in ARTable
  - Find interesting rules, dimensions, and categories in ARTable
  - Create and optimize the layout of ParSets
  - Validate interesting rules
  - Explore details of rules in ParSets using domain knowledge

Before

![Before](https://ars.els-cdn.com/content/image/1-s2.0-S2468502X1930021X-mmc1.mp4)

After

![After](https://ars.els-cdn.com/content/image/1-s2.0-S2468502X1930021X-mmc1.mp4)


ParSets displays dimensions as adjacent parallel axes and their values (categories) as segments over the axes (points in Parallel Coordinates [Inselberg, 2009]). Connections between categories in the parallel axes form ribbons (lines in parallel coordinates). The ribbon crossings lead to visual clutter.

Titanic example: Clutter 16.86% – alphabetical ordering of dimensions and categories.

Clutter 11.71% – ordering by the number of grades (categories) in each coordinate.

https://ars.els-cdn.com/content/image/1-s2.0-S2468502X1930021X-mmc1.mp4

Video
Visualizing dataflow graphs of deep learning models in Tensorflow

- **Problem:**
  - Difficulties to optimize *Deep Neural Networks* (DNN) models due to lack of understanding of DNN (*Black-box problem*).

- **Goals:** Visualization of DNN to
  - monitor learned parameters and output metrics
  - help training and optimizing DNN models.
  - help understand the dataflow graphs of DNN at the low-level.

- Monitoring, visualization and NN teaching tools:
  - Graph Visualizer,
  - TensorBoard, TensorFlow’s dashboard
  - Olah’s interactive essays
  - ConvNetJS,
  - TensorFlow Playground

- Modules for monitoring
  - scalar values, distribution of tensors, images, audio etc.

- Tools to visualize model structure at the higher-level than TensorFlow.

Visualizing dataflow graphs of deep learning models in Tensorflow

- While all dataflow tools are very useful, the major issue is that dataflow visualization itself *does not explain or optimize* the DNN model.

- An experienced **data scientist** should guide dataflow visualization for this.
  - *In contrast, the dataflow for explainable models can explain itself, as we can see for Decision Trees.*

- **Tracing** the movement of a given n-D point in the Decision Tree shows all interpretable decisions made to classify this point.
  - *For instance, consider a result of tracing 4-D point $\mathbf{x}=(7,2,4,1)$ in the DT through a sequence of nodes for attributes $x_3, x_2, x_4, x_1$ with the following thresholds: $x_3 < 5, x_2 > 0, x_4 < 5, x_1 > 6$ to a terminal node of class 1. The point $\mathbf{x}$ satisfies all these directly interpretable inequalities.*
Dataflow tracing in Decision Trees

(a) A traditional DT visualization for 9-D Wisconsin Breast Cancer (WBC) data clearly presents the structure of the DT model, but without explicitly tracing individual cases.

(b) The tracing is added with a dotted polyline. It shows two 5-D points \( a = (2.8, 5, 2.5, 5.5, 6.5) \) and \( b = (5, 8, 3, 4, 6) \).
   - Both points reach the terminal malignant edge of the DT, but with different certainty. The first point reaches it with lower certainty, having its values closer to the thresholds of coordinates denoted as \( uc \) and \( bn \).

- In this visualization, called Folded Coordinate Decision Tree (FC-DT) visualization, the edges of the DT not only connect decision nodes, but also serve as Folded Coordinates in disproportional scales for WBC data.

- Here, each coordinate is folded at the node threshold point with different lengths of the sides.

- For instance, with threshold \( T=2.5 \) on the coordinate \( uc \) with the interval of values \([1,10]\), the left interval is \([1, 2.5) \) and the right interval is \([2.5,10]\). In Figure b, these two unequal intervals are visualized with equal lengths, i.e., forming a disproportional scale.

DT dataflow tracing visualizations for WBC data.
Approaches to **discover** interpretable ML models aided by visual means beyond visualization of existing models

We are moving from *visualization of solution* to *finding solution visually*

- **Why Visual?**
  - To leverage human perceptual capabilities
- **Why interactive?**
  - To leverage human abilities to adjust tasks on the fly
- **Why Machine Learning?**
  - To leverage analytical discovery that are outside of human abilities.
  - We cannot see patterns in multidimensional data by a naked eye.
Approaches to **discover interpretable** ML models boosted by visual means beyond visualization of existing models

- **Components of approaches:**
  - *visual methods for n-D data representation*
  - *Visual methods for model discovery in visual n-D data representations*
  - *Methods to interpret data visual data representations and models that are not internally interpretable.*

- **Visual methods for 2-D/3-D representation of n-D data**
  - *Reversible/lossless/interpretable:* Parallel Coordinates, Radial Coordinates, General Line Coordinates, Shifted Paired Coordinates, Collocated Paired Coordinates, and others.
  - *Non-reversible/lossy/ with challenging interpretation:* PCA, MDF, RadVis, Manifolds, t-SNE and others
Visuals for creative thinking

- Scientists such as Bohr, Boltzmann, Einstein, Faraday, Feynman, Heisenberg, Helmholtz, Herschel, Kekule, Maxwell, Poincare, Tesla, Watson, and Watt have declared the fundamental role that images played in their most creative thinking. [Thagard & Cameron, 1997; Hadamard, 1954; Shepard & Cooper, 1982].
- *Albert Einstein: The words or the language, as they are written or spoken, do not seem to play any role in my mechanism of thought.*
Chinese and Indians knew a visual proof of the Pythagorean Theorem in 600 B.C. before it was known to the Greeks [Kulpa, 1994]. Below on the left

\[(a+b)^2 \text{ (area of the large square)} = a^2 + b^2 + ab + ab = (a+b)^2\]

\[a^2 + b^2 = (a+b)^2 \text{ (area of the large square) - 2ab (4 light green triangles)} = c^2 \text{ (area of inner darker green square)}\]

Thus, we follow this tradition -- moving from visualization of solution to finding solution visually with modern data science tools.
Example of visual knowledge discovery in 2-D

The common guess without visualizing data is to try a simplest linear discrimination function (black line) to separate the blue and red points. It will obviously fail.

In contrast a quick look at these data, immediately gives a visual insight of a correct model class of “crossing” two linear functions, with one line going over blue points, and another one going over the red points.

These “crossing” classes cannot be discriminated by a single straight line. Each single projection does not discriminates them. Projections overlap.
How to reproduce the success in 2-D for n-D data?

In high-dimensions we cannot see data with a naked eye. We need methods for lossless and interpretable visualization of n-D data in 2-D.
Multidimensional data visualization: Approaches

- The goal of **multivariate, multidimensional visualization** is representing **n-tuples** (n-D points) in 2-D or 3-D, e.g., (3, 1.0, 5, 7, 1.19, 0.13, 8.1)
- Often multidimensional data are visualized by lossy dimension reduction (e.g., PCA) or by splitting n-D data to a set of low dimensional data (pairwise correlation plots).
  - While splitting is useful it destroys integrity of n-D data, and leads to a shallow understanding complex n-D data.
- To mitigate splitting difficulty
  - an additional and difficult perceptual task of **assembling** 2-dimensional visualized pieces to the whole n-D record must be solved.
- An alternative way for deeper understanding of n-D data is
  - developing visual representations of n-D data in low dimensions without such data splitting and loss of information as graphs not 2-D points.
  - E.g., Parallel and Radial coordinates.
  - Challenge -- occlusion
WBC data

Benign and malignant cancer cases overlap. Interpretation of dimensions is difficult. Non-reversible lossy methods: 9-D to 2-D.

Fisher Discriminant Analysis – FDA

Wisconsin data set, top row: MDS and PCA, bottom row: FDA and SVM.

Theoretical limits to preserve n-D distances in 2-D: Johnson-Lindenstrauss Lemma

• This lemma implies that only a small number of arbitrary n-D points can be mapped to k-D points of a smaller dimension k that preserve n-D distances with relatively small deviations.

• Reason: the 2-D visualization space does not have enough neighbors with equal distances to represent the same n-D distances in 2-D.

• Result: the significant corruption of n-D distances in 2-D visualization.

Lemma [Johnson, Lindenstrauss, 1984].

Given $0 < \varepsilon < 1$, a set $X$ of $m$ points in $\mathbb{R}^n$, and a number $k > 8\ln(m)/\varepsilon^2$, there is a linear map $f : \mathbb{R}^n \rightarrow \mathbb{R}^k$ such that

$$(1 - \varepsilon) \| u - v \|^2 \leq \| f(u) - f(v) \|^2 \leq (1 + \varepsilon) \| u - v \|^2$$

for all $u, v \in X$.

In other words, this lemma sets up a relation between $n$, $k$ and $m$ when the distance can be preserved with some allowable error $\varepsilon$ for a linear mapping.
Version of lemma [Dasgupta, Gupta, 2003]

- Defines the possible dimensions $k < n$ such that for any set of $m$ points in $\mathbb{R}^n$ there is a mapping $f: \mathbb{R}^n \rightarrow \mathbb{R}^k$ with “similar” distances in $\mathbb{R}^n$ and $\mathbb{R}^k$ between mapped points. This similarity is expressed in terms of error $0 < \varepsilon < 1$.

- For $\varepsilon = 0$ these distances are equal. For $\varepsilon = 1$ the distances in $\mathbb{R}^k$ are less or equal to $\sqrt{2} S$, where $S$ is the distance in $\mathbb{R}^n$. This means that distance $s$ in $\mathbb{R}^k$ will be in the interval $[0, 1.42 S]$.

- In other words, the distances will not be more than 142% of the original distance, i.e., it will not be much exaggerated. However, it can dramatically diminish to 0. The exact formulation of this version of the Johnson-Lindenstrauss lemma is as follows.

- **Theorem 1** [Dasgupta, Gupta, 2003, theorem 2.1]. For any $0 < \varepsilon < 1$ and any integer $n$, let $k$ be a positive integer such that

$$k \geq 4(\varepsilon^2 / 2 - \varepsilon^3 / 3)^{-1} \ln n \quad (3.1)$$

then for any set $V$ of $m$ points in $\mathbb{R}^k$ there is a mapping $f: \mathbb{R}^n \rightarrow \mathbb{R}^k$ such that for all $u, v \in V$

$$|1 - \varepsilon| ||u-v||^2 \leq ||f(u)-f(v)||^2 \leq (1 + \varepsilon) ||u-v||^2 \quad (3.2)$$
Theoretical limits to preserve n-D distances in 2-D: Johnson-Lindenstrauss Lemma

Johnson-Lindenstrauss Lemma shows that to keep distance errors within about 30% for just 10 arbitrary high-dimensional points, we need over 1900 dimensions, and over 4500 dimensions for 300 arbitrary points. Visualization methods do not meeting these requirements.
Review of Visual Knowledge Discovery Approaches

1. Convert n-D data to 2-D data and then discover **2-D patterns** in visualization as **points**.

2. Discover **n-D patterns** in n-D data then visualize them in 2-D as **graphs**.

3. Join 1 and 2: some patterns are discovered in (1) with controlled errors and some are discovered in (2).
1. Convert n-D data to 2-D data and then discover 2-D patterns in visualization.

- **Lossy approach**
  - Lossy conversion to 2-D (dimension reduction, DR)
  - Point to point (n-D point to 2-D point)
  - Visualization in 2-D
  - Interactive discovery of 2-D patterns in visualization

- **Lossless approach**
  - Lossless conversion (visualization) to 2-D (n-D data fully restorable from its visualization) Point to graph (n-D point to graph in 2-D), e.g., (5,4,0,6,5,10) to a graph
  - Interactive discovery of 2-D patterns on graphs in visualization
Examples: Example 1: linear discrimination of 4-D data, n-D point to graph

Algorithm GLC-L
• 4 coordinate lines: \( X_1, X_2, X_3, X_4 \) -- black lines (vectors) at different angles \( Q_1-Q_4 \)
• 4-D point \((x_1, x_2, x_3, x_4) = (1, 0.8, 1.2, 1)\) with these values shown as blue lines (vectors)
• Shifting and stacking blue lines
• Projecting the last point to U line
• Do the same for other 4-D points of blue class
• Do the same for 4-D points of red class
• Optimize angles \( Q_1-Q_4 \) to separate classes (yellow line)
Example 2: Visual linear discrimination of 9-D Wisconsin Breast Cancer data in GLC-L coordinates

- Critical in Medical diagnostics and many other fields:
  - Explanation of patterns and
  - Understanding them
- Lossless visual means
- Reversible/restorable

Only one malignant (red case) on the wrong side

Angles $Q_1$--$Q_9$ -- contributions of attributes $X_1$--$X_9$

<table>
<thead>
<tr>
<th>Real Class</th>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>424</td>
</tr>
<tr>
<td>Class 2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>238</td>
</tr>
</tbody>
</table>

Accuracy is: 96.9253%
Example 3: Visual discrimination of 9-D Wisconsin Breast Cancer data in Shifted Paired Coordinates (SPC)

Point a in $(X_1,X_2), (X_3,X_4), (X_5,X_6)$ as a sequence of pairs (3,2), (1,4) and (2,6).

Point a in $(X_2,X_1), (X_3,X_4), (X_5,X_6)$ as a sequence of pairs (2,3), (1,6) and (2,4).

A set of 688 Wisconsin Breast Cancer (WBC) data visualized in SPC as 2-D graphs of 10-D points with benign cases in red and malignant cases in blue.

Here malignant cases (blue cases)

WBC cases covered by else in rule 2 (dominated by blue class).

WBC data in 4-D SPC as graphs in coordinates $(X_3,X_4)$ and $(X_6,X_7)$ that are used by Rule 1, i.e., WBC cases that go through $R_1$ and not go to $R_2$ and $R_3$ in these coordinates.

The WBC explainable classification **Rule 2** is

If $(x_8,x_9) \in R_1$ & $(x_6,x_7) \notin R_2$ & $(x_6,x_7) \notin R_3$

then $x \in$ class 1 (Red, Benign) else $x \in$ class 2 (Blue, Malignant)

This rule classifies all cases that are either in $R_2$ or in $R_3$ or not in $R_1$ as blue.

It has accuracy 93.60% (425+219)/688) on all 688 cases
Example 4: Avoiding Occlusion with Deep Learning on WBC data

Numeric 9-D Wisconsin Breast Cancer (WBC) Data

Visualized numeric data as downscaled 25x25 pixels images using GLC-L method

Deep Learning Convolutional Neural Network (CNN) on images (GLC-L visualization) as input

Classification accuracy 97.22% at the level and above published in literature

WBC data samples visualized in GLC-L for CNN model with the best accuracy.

Visualization optimization

10-fold cross validation
EXAMPLE 5: INTERPRETABLE DCP ALGORITHM ON WBC DATA

1. Constructing dominance intervals for classes on each attribute
2. Combining dominance intervals in the voting methods
3. Learning parameters of dominance intervals and voting methods for prediction
4. Visualizing the dominance structure
5. Explaining the prediction

Average accuracy of 10-fold cross validation is 97.01%
Improved algorithm 99.3%
General Line Coordinates (GLC) to convert n-D points to graphs in 2-D
General Line Coordinates (GLC)  

Descartes lay in bed and invented the method of co-ordinate geometry.  

Alfred North Whitehead

7-D point \( D=(5,2,5,1,7,4,1) \) in Parallel Coordinates

(b) 7-D point \( D \) in General Line Coordinates with curvilinear lines.

(c) 7-D points \( F-J \) in General Line Coordinates that form a simple single straight line.

(d) 7-D points \( F-J \) in Parallel Coordinates that do not form a simple single straight line.

7-D points in General Line Coordinates with different directions of coordinates \( X_1, X_2, \ldots, X_7 \) in comparison with Parallel Coordinates.
General Line Coordinates (GLC)

n-Gon (rectangular) coordinates with 6-D point (0.5, 0.6, 0.9, 0.7, 0.7, 0.1).

3-D point A=(0.3, 0.7, 0.4) in 3-Gon (triangular) coordinates and in radial coordinates.

(a) Point A in 3-Gon coordinates.  
(b) Point A in in radial coordinates.

(a) Parallel Coordinates display.  
(b) Circular Coordinates display.  
(c) Spatially distributed objects in circular coordinates with two coordinates X_5 and X_6 used as a location in 2-D and X_7 is encoded by the sizes of circles.

Figure 2.5. Examples of circular coordinates in comparison with parallel coordinates.

(a)  Example in n-Gon coordinates with curvi-linear edges of a graph.  
(b) Example in n-Gon coordinates with straight edges of a graph.

Figure 2.6 Example of weekly stock data in n-Gon (pentagon) coordinates.
General Line Coordinates (GLC): 2-D visualization

2-D Line Coordinates.

<table>
<thead>
<tr>
<th>Type</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-D General Line Coordinates (GLC)</td>
<td>Drawing n coordinate axes in 2-D in variety of ways: curved, parallel, unpar-</td>
</tr>
<tr>
<td></td>
<td>allel, collocated, disconnected, etc.</td>
</tr>
<tr>
<td>Collocated Paired Coordinates (CPC)</td>
<td>Splitting an n-D point x into pairs of its coordinates ((x_1,x_2),\ldots,(x_{n-1},x_n)); drawing each pair as a 2-D point in the collocated axes; and linking these points to form a directed graph. For odd n coordinate (X_n) is repeated to make (n) even.</td>
</tr>
<tr>
<td>Basic Shifted Paired Coordinates (SPC)</td>
<td>Drawing each next pair in the shifted coordinate system by adding ((1,1)) to the second pair, ((2,2)) to the third pair, ((i-1, i-1)) to the (i)-th pair, and so on. More generally, shifts can be a function of some parameters.</td>
</tr>
<tr>
<td>2-D Anchored Paired Coordinates (APC)</td>
<td>Drawing each next pair in the shifted coordinate system, i.e., coordinates shifted to the location of a given pair (anchor), e.g., the first pair of a given n-D point. Pairs are shown relative to the anchor easing the comparison with it.</td>
</tr>
<tr>
<td>2-D Partially Collocated Coordinates (PCC)</td>
<td>Drawing some coordinate axes in 2D collocated and some coordinates not col-</td>
</tr>
<tr>
<td>In-Line Coordinates (ILC)</td>
<td>Drawing all coordinate axes in 2D located one after another on a single straight line.</td>
</tr>
<tr>
<td>Circular and n-Gon coordinates</td>
<td>Drawing all coordinate axes in 2D located on a circle or an n-Gon one after another.</td>
</tr>
<tr>
<td>Elliptic coordinates</td>
<td>Drawing all coordinate axes in 2D located on ellipses.</td>
</tr>
<tr>
<td>GLC for linear functions (GLC-L)</td>
<td>Drawing all coordinates in 2D dynamically depending on coefficients of the linear function and value of (n) attributes.</td>
</tr>
<tr>
<td>Paired Crown Coordinates (PWC)</td>
<td>Drawing odd coordinates collocated on the closed convex hull in 2-D and even coordinates orthogonal to them as a function of the odd coordinate.</td>
</tr>
</tbody>
</table>
# General Line Coordinates (GLC): 3-D visualization

3-D Line Coordinates.

<table>
<thead>
<tr>
<th>Type</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-D General Line Coordinates (GLC)</td>
<td>Drawing ( n ) coordinate axes in 3-D in variety of ways: curved, parallel, unparallelled, collocated, disconnected, etc.</td>
</tr>
<tr>
<td>Collocated Tripled Coordinates (CTC)</td>
<td>Splitting ( n ) coordinates into triples and representing each triple as 3-D point in the same three axes; and linking these points to form a directed graph. If ( n \mod 3 ) is not 0 then repeat the last coordinate ( X_n ), one or two times to make it 0.</td>
</tr>
<tr>
<td>Basic Shifted Tripled Coordinates (STC)</td>
<td>Drawing each next triple in the shifted coordinate system by adding ((1,1,1)) to the second triple, ((2,2,2)) to the third triple ((i-1, i-1, i-1)) to the ( i )-th triple, and so on. More generally, shifts can be a function of some parameters.</td>
</tr>
<tr>
<td>Anchored Tripled Coordinates (ATC) in 3-D</td>
<td>Drawing each next triple in the shifted coordinate system, i.e., coordinates shifted to the location of the given triple of (anchor), e.g., the first triple of a given n-D point. Triple are shown relative to the anchor easing the comparison with it.</td>
</tr>
<tr>
<td>3-D Partially Collocated Coordinates (PCC)</td>
<td>Drawing some coordinate axes in 3-D collocated and some coordinates not collocated.</td>
</tr>
<tr>
<td>3-D In-Line Coordinates (ILC)</td>
<td>Drawing all coordinate axes in 3D located one after another on a single straight line.</td>
</tr>
<tr>
<td>In-Plane Coordinates (IPC)</td>
<td>Drawing all coordinate axes in 3D located on a single plane (2-D GLC embedded to 3-D).</td>
</tr>
<tr>
<td>Spherical and polyhedron coordinates</td>
<td>Drawing all coordinate axes in 3D located on a sphere or a polyhedron.</td>
</tr>
<tr>
<td>Ellipsoidal coordinates</td>
<td>Drawing all coordinate axes in 3D located on ellipsoids.</td>
</tr>
<tr>
<td>GLC for linear functions (GLC-L)</td>
<td>Drawing all coordinates in 3D dynamically depending on coefficients of the linear function and value of ( n ) attributes.</td>
</tr>
<tr>
<td>Paired Crown Coordinates (PWC)</td>
<td>Drawing odd coordinates collocated on the closed convex hull in 3-D and even coordinates orthogonal to them as a function of the odd coordinate value.</td>
</tr>
</tbody>
</table>
Reversible Lossless Paired Coordinates

State vector \( \mathbf{x} = (x,y,x',y',x'',y'') = (0.2, 0.4, 0.1, 0.6, 0.4, 0.8) \) in Collocated Paired and Parallel Coordinates.
Reversible lossless Paired Coordinates

6-D point as a closed contour in 2-D where a 6-D point \(x=(1, 1, 2, 2, 1, 1)\) is forming a tringle from the edges of the graph in Paired Radial Coordinates with non-orthogonal Cartesian mapping.

6-D point \((1, 1, 1, 1, 1, 1)\) in two \(X_1-X_6\) coordinate systems (left – in Radial Collocated Coordinates, right – in Cartesian Collocated Coordinates).

n-D points as closed contours in 2-D: (a) 16-D point \((1,1,2,2,1,1,2,1,1,2,1,1,2,1,2)\) in Partially Collocated Radial Coordinates with Cartesian encoding, (b) CPC star of a 192-D point in Polar encoding, (c) the same 192-D point as a traditional star in Polar encoding.

4-D point \(P=(0.3,0.5,0.5,0.2)\) in 4-D Elliptic Paired Coordinates, EPC-H as a green arrow. Red marks separate coordinates in the Coordinate ellipse.

4-D point \(P=(0.3,0.5,0.5,0.2)\) in Radial Coordinates.
Graph construction algorithms in GLC

Six coordinates and six vectors that represent a 6-D data point (0.75, 0.5, 0.7, 0.6, 0.7, 0.3) in GLC-SC1.

6-D data point (0.75, 0.5, 0.7, 0.6, 0.7, 0.3) in GLC-CC1.

6-D data point (0.75, 0.5, 0.7, 0.6, 0.7, 0.3) in GLC-PC.

6-D data point (0.75, 0.5, 0.7, 0.6, 0.7, 0.3) in GLC-CC2.

6-D data point (0.75, 0.5, 0.7, 0.6, 0.7, 0.3) in GLC-SC1. 6-D data point (0.75, 0.5, 0.7, 0.6, 0.7, 0.3) in GLC-SC2.
Math, theory and pattern simplification methodology: Statements

- **Statement 1.** Parallel Coordinates, CPC and SPC preserve $L^p$ distances for $p=1$ and $p=2$, $D(x,y) = D^*(x^*,y^*)$.

- **Statement 2** ($n$ points lossless representation). If all coordinates $X_i$ do not overlap then GLC-PC algorithm provides bijective 1:1 mapping of any n-D point $x$ to 2-D directed graph $x^*$.

- **Statement 3** ($n$ points lossless representation). If all coordinates $X_i$ do not overlap then GLC-PC and GLC-SC1 algorithms provide bijective 1:1 mapping of any n-D point $x$ to 2-D directed graph $x^*$.

- **Statement 4** ($n/2$ points lossless representation). If coordinates $X_i$ and $X_{i+1}$ are not collinear in each pair $(X_i, X_{i+1})$ then GLC-CC1 algorithm provides bijective 1:1 mapping of any n-D point $x$ to 2-D directed graph $x^*$ with $\lceil n/2 \rceil$ nodes and $\lceil n/2 \rceil - 1$ edges.

- **Statement 5** ($n/2$ points lossless representation). If coordinates $X_i$ and $X_{i+1}$ are not collinear in each pair $(X_i, X_{i+1})$ then GLC-CC2 algorithm provides bijective 1:1 mapping of any n-D point $x$ to 2-D directed graph $x^*$ with $\lceil n/2 \rceil$ nodes and $\lceil n/2 \rceil - 1$ edges.
Math, theory and pattern simplification methodology: Statements

- **Statement 6** (n points lossless representation). If all coordinates $X_i$ do not overlap then GLC-SC2 algorithm provides bijective 1:1 mapping of any n-D point $x$ to 2-D directed graph $x^*$.

- **Statement 7**. GLC-CC1 preserves $L^p$ distances for $p=1$, $D(x,y) = D^*(x^*,y^*)$.

- **Statement 8**. In the coordinate system $X_1, X_2, \ldots, X_n$ constructed by the Single Point algorithm with the given base n-D point $x=(x_1, x_2, \ldots, x_n)$ and the anchor 2-D point $A$, the n-D point $x$ is mapped one-to-one to a single 2-D point $A$ by GLC-CC algorithm.

- **Statement 9** (locality statement). All graphs that represent nodes $N$ of n-D hypercube $H$ are within square $S$.
Adjustable GLCs for decreasing occlusion and pattern simplification

Simplicity is the ultimate sophistication.
Leonardo da Vinci

Non-preattentive vs. preattentive visualizations (linearized patterns): 6-D point A= (3, 6, 4, 8, 2, 9) in blue, and 6-D point B=(3.5, 6.8, 4.8, 8.5, 2.8, 9.8) in orange in Traditional, Shifted Parallel Coordinates, and GLC.
Case Studies: World Hunger data

4-D data: representation of prevalence of undernourished in the population (%) in Collocated Paired Coordinates

4-D data: representation of prevalence of undernourished in the population (%) in traditional time series (equivalent to Parallel Coordinates for time series)

The Global Hunger Index (GHI) for each country measures as,

\[ \text{GHI} = \frac{(\text{UNN} + \text{UW5} + \text{MR5})}{3}, \]

where UNN is the proportion of the population that is Undernourished (in %), UW5 is the prevalence of Underweight in children under age of five (in %), and MR5 is the Mortality rate of Children under age five (in %).
Case Studies: Health Monitoring with PC and CPC

The colors show the progress to the goal.
- Dark green dot – goal.
- Yellow and light green – closer to the goal point.
- Red arrow – initial health status.

- Experiments -- people quickly grasp how to use this health monitor.
- This health monitor is expandable.
- Two more indicators is another pair of shifted Cartesian Coordinates.
  - The goal is the same dark green 2-D dot
  - Each graph has two connected arrows.
- Graphs closer to the goal are smaller.
Case studies: Knowledge Discovery and Machine Learning for Investment Strategy with CPC

- The CPC visualization shows arrows in \((V_r,Y_r)\) space of volume \(V_r\) and relative main outcome variable \(Y_r\).
- This is a part of the data shown as traditional time series with time axis.
- CPC has no time axis. The arrow direction shows time.
- The arrow beginning is the point in the space \((V_r,Y_r)\), and its head is the next time point in the collocated space \((V_{r+1},Y_{r+1})\).
- CPC give the inspiration idea for building a trading strategy in contrast with time series figure without it.
  - It allows finding the areas with clusters of two kinds of arrows.
  - The arrows for the long positions are green arrows.
  - The arrows for the short positions, are red.
  - Along the \(Y_r\) axis we can observe a type of change in \(Y\) in the current candle. if \(Y_{r+1}>Y_r\) then \(Y_{r+1}>Y_r\) the right decision in \(i\)-point is a long position opening. Otherwise, it is a short position.
  - Next, CPC shows the effectiveness a decision in the positions.
  - The very horizontal arrows indicates small profit
  - A more vertical arrows indicates the larger profit.
- In comparison with traditional time series, the CPC bring the additional knowledge about the potential of profit in selected area of parameters in \((V_r,Y_r)\) space.
Figure 8.16. Pins in 3-D space: two cubes found in \((Y_r, dMA_r, V_r)\) space with the maximum asymmetry between long and short positions.

Figure 8.17. The zoomed cubes with the best asymmetry from Figure 8.16. The upper cube with green circles is selected for long positions lower cube with red circles is for short positions. For better visibility, the viewpoint is changed from Figure 8.16.

Figure 8.18. Two determined cubes in \(Y_r-dMA_r-V_r\) space with the maximum asymmetry between long and short positions for the new grid resolution.
Visual Text Mining: Discovery of Incongruity in Humor Modeling

Two fish are in the tank. One looks to the other and asks: “How do you drive this thing?”

Incongruity process for model $M$.

<table>
<thead>
<tr>
<th>Time</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>Agent $G$ (human or a software agent) reads the first part of the text $P_1$ and concludes (at a surface level) that $P_1$ is a usual text.</td>
</tr>
<tr>
<td>$t_2$</td>
<td>Agent $G$ reads the second part of the text $P_2$ and concludes that (at a surface level) that $P_2$ is a usual text.</td>
</tr>
<tr>
<td>$t_3$</td>
<td>Agent $G$ starts to analyze a relation between $P_1$ and $P_2$ (at a deeper semantic level). Agent $G$ retrieves semantic features (words, phrases) $F(P_1)$ of $P_1$.</td>
</tr>
<tr>
<td>$t_4$</td>
<td>Agent $G$ retrieves semantic features (words, phrases) $F(P_2)$ of $P_2$.</td>
</tr>
<tr>
<td>$t_5$</td>
<td>Agent $G$ compares (correlates) features $F(P_1)$ with $P_2$ and features $F(P_2)$ with $P_1$ finding significant differences in meaning (incongruity).</td>
</tr>
<tr>
<td>$t_6$</td>
<td>Agent $G$ reevaluate usuality of $P_2$ taking in to account these correlations and concludes that $P_2$ is unusual.</td>
</tr>
</tbody>
</table>

Confusion matrix for visual classification rule R1

| Actual jokes | 17 | 15 | 2 |
| Actual non-jokes | 17 | 2 | 15 |

We record this visual discovery as a rule R2:

R2: If $z_3 < z_4$ then $z$ is a joke, else $z$ is a non-joke.

The resulting C4.5 rule R3 is

If $z_3 < z_4 < 0.0075$ then $z$ is a joke,
else $z$ is a non-joke.

4 attributes that express semantic correlations of Part 1 & Part 2 of the joke

Collocated Paired Coordinate (CPC) plot of meaning context correlation over time. The set of jokes and non-jokes plotted as meaning correlation over time.

Monotone Boolean space of jokes (green dots) and non-jokes (red dots).

R4: If $(w \geq (1,0,1,1) \lor w \geq (0,1,0,1)$ then $w$ is joke else $w$ is a non-joke.
Case study: Recognition of digits with dimension reduction

Cropped input Images 22x22 = 484 pixels cropped edges from 784 pixels
From 784-D to 484-D then to 249-D by GLC-L algorithms.

Experiments with 900 samples of MNIST dataset for digits 0 and 1. Results of the automatic dimension reduction displaying 249 dimensions with 235 dimensions removed with the accuracy dropped by 0.28%.

Experiments with 900 samples of MNIST dataset for digits 0 and 1. Results for the best linear discriminant function of the first run of 20 epochs in 484-D.
Case study: Recognition of digits

Dimension reduction with NN Autoencoder

**Lossy but controlled Dimension Reduction** with NN autoencoder

- Input layer = Output layer
- 484 nodes each
- Hidden layer has 24 nodes

A subset of the testing set between 50 to 100 samples for each digit
- 24-dimensional data with 10 different classes
- 24/784 is 3% after preprocessing and decoding

Cropped input Images 22x22 = 484 pixels
cropped edges from 784 pixels

Decoded images with hidden Layer: 24 nodes

0  2  6  5  8  9
Encoded digit 0 and digit 1 on and GLC-L, using 24 dimensions found by the Autoencoder among 484 dimensions. Results for the best linear discriminant function of the first run of 20 epochs.

Encoded digit 0 and digit 1 on GLC-L, using 24 dimensions found by the Autoencoder among 484 dimensions. Another run of 20 epochs, best linear discriminant function from this run. Accuracy drops 1%.

Encoded digit 0 in Parallel Coordinates using 24 dimensions found by the Autoencoder among 484 dimensions. Each vertical line is one of the 24 features scaled in the [0,35] interval. Digit 0 is visualized in Parallel Coordinates.

Encoded digit 1 in Parallel Coordinates using 24 dimensions found by the Autoencoder among 484 dimensions. Each vertical line is one of the 24 features scaled in the [0,35] interval. Digit 1 is visualized on the parallel coordinates.
Visual rule design from embedding generated by auto-encoder

- How to classify 0 and 1 using these visualizations?
  - Select discriminating features.
  - Black circles show the differences between 0 and 1 digits in these 24 coordinates.
  - Design rules and explain them, e.g.,
    if $x_2 = 0$ then 0; if $x_{18} = 0$ then 0; if $x_7 = 0$ then 0.
    If $x_{24}$ is in green oval then 0.
  - Remove cases that satisfy these simple rules
  - Search rules form remaining cases

- How to interpret these 24 features?
  - Tracing these 24 features to find their origin in the image.
Collaborative visual discovery

Data and Task Split-based Collaborative Visualization framework.

Task splitting for Collaborative Visualization.

Experiments: success in dimension n>100
Current Software
Summary on General Line Coordinates

- The examples and cases studies show that hybrid methods with General Line Coordinates are capable
  - visualizing data of multiple dimensions from 4-D to 484-D without loss of information and
  - discovering patterns by combining humans perceptual capabilities and Machine Learning / Data Mining algorithms for classification such high-dimensional data.

- This Hybrid technique can be developed further in multiple ways to deal with different new challenging data science tasks.
Approaches to explain deep and other ML models by visual means

- Explaining ML models including deep learning models by visual means
  - activation and weight visualization,
  - heatmap-based methods,
  - dependency analysis,
  - monotonicity approach – monotone Boolean functions and chains;
  - decision tree visualization, and others.
Tools for explaining deep learning models

- Tools that **visualize activations** generated on every layer of a trained convolutional net during image/video recognition.
  - **Benefits**: builds intuitions how the network work (**algorithm tracing**).

- Tool that **visualize features** at every layer of a network discovered by optimization in image space.
  - **Challenge** --- **blurred** and less recognizable images of features.
  - **Approach**: new **regularization methods** for feature optimization to generate clearer, more interpretable visualizations.

- A new term -- **deep visualization**

---

Re visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]
Visualization of Image features in heat map

CIFAR-10 classification benchmark problem is to classify RGB 32x32 pixel images across 10 categories.

CIFAR-10 is a multi-layer network with alternating convolutions and nonlinearities followed by fully connected layers and softmax classifier.

Peak accuracy \( \sim 86\% \) few hours of training on a GPU.

\( \sim 1M \) learnable parameters

19.5M multiply-add operations to compute inference on a single image.

Discovering human explainable visualization of what a deep neural network has learned

- Comparison of three heatmaps for digit ‘3’.
- Left: The randomly generated heatmap – no interpretable information.
- Middle: The segmentation heatmap – shows the whole digit without relevant parts, say, for distinguishing ‘3’ from ‘8’ or ‘9’.
- Right: A relevance heatmap – shows parts of the image used by the classifier. Reflects human intuition on differences between ‘3’, ‘8’ and ‘9’ and other digits.

Comparison of the three heatmap computations

- **Left:**
  - **Sensitivity heatmaps** (local explanations) – measure the change of the class when specific pixels are changed based on partial derivatives.
  - applicable to architectures with differentiable units.

- **Middle:**
  - **Deconvolution method** (“autoencoder”) – applies a convolutional network $g$ to the output of another convolutional network $f$. Network $g$ “undoes” $f$.
  - Challenge: Unclear relation between heatmap scores and the classification output $f(x)$.

- **Right:**
  - **Layer-wise Relevance Propagation** (LRP) – exactly decomposes the classification output $f(x)$ into pixel relevancies by observing the layer-wise evidence for class preservation (conservation principle).
  - Applicable to generic architectures (including with non-continuous units) – does not use gradients.
  - Globally explains the classification decision and heatmap scores
  - Have a clear interpretation by a human as evidence for or against a class.

Multifaceted Feature Visualization

- The studies with “fooling” images had shown that trained DNNs
  - ignore an object’s global structure, and instead only
  - learn a few, discriminative features per class (e.g. color or texture) (Nguyen et al., 2015).

- Multifaceted Feature Visualization algorithm:
  - shows the multiple feature facets each neuron detects.

- Benefits
  - more comprehensive understanding of each neuron’s function.
  - Opens opportunity to create more powerful deep learning algorithms.


Learning deep features for scene recognition using places database

- **Challenge:**
  - performance at scene recognition is lower than for object recognition.

- **Reasons:**
  - Current deep features trained from ImageNet are not competitive enough for such tasks.

- **Approach:**
  - scene-centric database called Places with over 7 million labeled pictures of scenes.
  - methods to compare the **density and diversity** of image datasets
  - **CNN** to learn deep features for scene recognition
  - **Heatmap Visualization** of the CNN layers’ responses to show differences in the internal representations of object-centric and scene-centric networks.

Heat map visualization in identification of diabetic retinopathy using deep learning

- A common method of combining results of Deep Learning (DL) from images with visualization is:
  - discovering classification model for images using a DL algorithm,
  - identifying informative deep features (inner features constructed by this DL algorithm), and
  - visualizing identified deep features on the original image.

- The methods of visualization of these features range from
  - outlining the area of deep features to
  - overlaying the heat map in these areas (B in the image).

- Issue – Are visualized features always explainable?

Figure 5. Visualization maps generated from deep features. A. Fundus heatmap overlaid on a fundus image, highlighting pathological regions in the nasal and temporal quadrants. B. Pathologic findings in the upper and lower left quadrants. These visualizations are generated automatically, locating regions for closer examination after a patient is seen by a consultant ophthalmologist.

Explanatory Interactive Learning (XIL) for Deep Networks

- DNN can use **confounding** factors within datasets to achieve **high prediction** of the trained models.
  - These factors can be good predictors in a given dataset, but be **useless in real world** settings [Lapuschkin et al., 2019].
  - For instance, the model can be right in prediction, but for the **wrong reasons**, focusing incorrectly on areas outside of the tissue of interest.

- Options:
  - **Discard** such models and datasets
  - **Correct** such models by the human user interactively [Schramowski et al., 2020]
    - by adding more and better training cases - a user provides **counterexamples** that teach the learner not to depend on the irrelevant components.
    - by adding **annotated masks** during the learning loop and
    - by **penalizing** decisions made for wrong reasons.
  - Result: focusing on **relevant features**, without considerably dropping predictive performance.

Explanatory Interactive Learning (XIL)

- The additional regularization term function adds a penalty to gradients that lie outside of a binary mask that indicates which features of the input are relevant.

\[
X, \ y, \ A = \sum_{n=1}^{N} \sum_{k=1}^{K} -c_k y_{nk} \log(\hat{y}_{nk}) + \lambda_1 \sum_{n=1}^{N} \sum_{d=1}^{D} \left( A_{nd} \delta \sum_{k=1}^{K} c_k \log(\hat{y}_{nk}) \right)^2 + \lambda_2 \sum_{i}^{t} \theta_i^2
\]

- Experiment with decoy variant of MNIST images of 10 digits (70,000 images).
  - All images are corrupted by introducing confounders, 44 patches of pixels in randomly chosen corners whose shade is a function of the label in the training set and random in the test set.
  - The average test set accuracy of a multilayer perceptron for 3 strategies: no corrections - 48%; counterexample strategy (CE) – 82% for a single counterexample.; the input-gradient constraints - 85%

<table>
<thead>
<tr>
<th></th>
<th>no corr.</th>
<th>Counterexamples</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>c = 1</td>
<td>c = 3</td>
</tr>
<tr>
<td></td>
<td>c = 5</td>
<td>RRR</td>
</tr>
<tr>
<td>Train</td>
<td>97%</td>
<td>93%</td>
</tr>
<tr>
<td></td>
<td>92%</td>
<td>92%</td>
</tr>
<tr>
<td></td>
<td>89%</td>
<td>99%</td>
</tr>
<tr>
<td>Test</td>
<td>48%</td>
<td>82%</td>
</tr>
<tr>
<td></td>
<td>85%</td>
<td>85%</td>
</tr>
<tr>
<td></td>
<td>85%</td>
<td></td>
</tr>
</tbody>
</table>

- Heatmaps

- Experiment with plant phenotyping data, CNN accuracy
  - no corrections – 88-99%, but significantly for the wrong reasons,
  - the input-gradient constraints 88-95% more for the right reason

A user cannot review thousands of images on correctness of heatmaps in training and validation data. This review process is not scalable to thousands of images.
What catches the eye? Deep saliency models vs. human gaze.

- Better performing DL models have higher proportions of deep neurons highly predictive of human gaze.
- The predictive neurons are attuned to clear semantic categories such as animals (dogs, cats), objects (motorbike, ball) and parts (head, hair).
- This hints that saliency, as experienced by humans, likely involves high-level world knowledge in addition to low-level perceptual cues.
- Computational approach to improve DL: minimizing the distance between the predicted saliency maps and the ground truth recorded by human gaze.
- Several saliency prediction datasets are currently available in literature. The largest one is SALICON composed by 20,000 images with corresponding saliency maps computed from mouse movements.


Unsupervised feature learning for audio classification using Convolutional Deep Belief Network (CDBN)

- **Computational Approach**
  - Convert time-domain signals into spectrograms.
  - Extract spectrogram with overlaps from each TIMIT training data case.
  - Reduce dimension of the spectrogram by PCA to 80 principal components.
  - Training first-layer cases.
  - Training second-layer cases using first-layer activations as input.

- **Visualization Approach**
  - Finding what the network “learns” using visualization.
    - Visualizing the first layer bases
    - Visualizing each second-layer learned representation as a weighted linear combination of the first-layer bases.

A **spectrogram** is a map image with axes for time and frequency and intensity or color of a point for amplitude.

**TIMIT** — recordings of 630 speakers of eight major dialects of American English, each reading ten phonetically rich sentences.

Heat maps visualization and explanation of deep learning for pulmonary tuberculosis

Chest radiograph with pathologically proven active TB.

The same radiograph with a heat map overlay of the strongest activations from the 5th convolutional layer from GoogLeNet-TA classifier.

The red and light blue regions in the upper lobes — areas activated by the deep neural network. (areas where the disease is present)
The dark purple background — areas that are not activated.

Generative Adversarial Networks (GANs) visualization

- In GAN the generative network generates candidates while the discriminative network evaluates them, however, visualization and understanding of GANs is largely missing.
- How does a GAN represent our visual world internally?
- What causes the artifacts in GAN results?
- How do architectural choices affect GAN learning?
- A framework to visualize and understand GANs at the unit-, object-, and scene-level.
- Step 1: identify interpretable units closely related to object concepts with a segmentation-based network dissection.
- Step 2: quantify their causal effect by measuring interventions to control objects in the output.
- Step 3: examine the contextual relationship between these units and their surrounding by inserting the discovered objects into new images.

Inserting door units by setting 20 causal units

- Whether the door units can cause the generation of doors is dependent on its local context.
- The idea is to insert extra pixels (door pixels). The opposite common idea is covering some pixels to find salient pixels.

Nonlinear feature space dimension reduction in breast computer-aided diagnosis (CADx) with t-SNE

- **Goals:**
  - Enhancing breast CADx with unsupervised dimension reduction (DR)
  - Representation of computer-extracted breast lesion feature spaces for
    - 1126 ultrasound cases,
    - 356 MRI cases,
    - 245 mammography cases.
- **DR methods:**
  - \( t \)-distributed stochastic neighbor embedding (t-SNE) – probability distribution based
- **Intension of DR methods:**
  - Convert high dimensional feature spaces to more human interpretable lower dimensional spaces preserving local and global information.
- **Malignancy classifiers:**
  - Markov chain Monte Carlo based Bayesian artificial neural network (MCMC-BANN)
  - and Linear Discriminant Analysis (LDA).

- **Evaluation criteria:**
  - Performance of classifiers with DR relative to malignancy classification (using receiver operating curve (ROC) analysis, 95% empirical confidence intervals for each classifier’s area under curve (AUC) for ROC curves).
  - Visual inspection of sparseness of 2-D and 3-D mappings.
  - Comparison with previous breast CADx (automatic relevance determination and linear stepwise (LSW) feature selection, a linear DR based on PCA, and Automatic relevance determination (ARD) feature selection)

- **Results:**
  - Slightly higher performance of 4-D t-SNE mapping (from the original 81-D feature space) relative to 13 ARD selected features and 4 LSW selected features for large ultrasound dataset using the MCMC-BANN classifier.

- **Conclusion:**
  - DR techniques offer a complementary approach to known tools.
  - Added benefit – delivering sparse lower dimensional representations for visual interpretation,
  - revealing intricate data structure of the feature space ???.

Lossy visualizations for ML models

- 2D and 3D visualizations of \textit{unsupervised} reduced dimension representations of 81-D breast lesion ultrasound feature data
  - green – Benign lesions, Red - Malignant, Yellow – Benign-cystic.

- Visualization of linear reduction using
  - (a) PCA, first two principal components
  - (b) first three principal components, 3D PCA.
  - (c) 2D and (d) 3D visualization of the nonlinear reduction mapping using t-SNE
  - (e) 2D and (f) 3D visualization of the nonlinear mapping using Laplacian eigenmaps.

- Discussion:
  - Methods like T-SNE, PCA and others do not preserve all information of initial features (they are lossy visualizations of n-D data)
  - They convert 81 interpretable features to 2-3 artificial features that have no direct interpretation
  - General Line Coordinates is an alternative that preserves all n-D information when occlusion/clutter in visualization is suppressed that was successfully done in [Kovalerchuk et al, 2014-2019]

T-SNE visualization for explainable AI in intrusion detection systems: an adversarial approach

- **Goal:**
  - Increase understanding of “black-box” Deep Neural Networks (DNN) models.

- **Approach:**
  - Explain *miss-classifications* made by Intrusion Detection Systems
  - Find *minimum modifications* (of the input features) to *correct classification* of misclassified cases by using adversarial machine learning
  - Make intuitive visualization of magnitude of max modifications as *most explanatory features* for the misclassification.

- **Results:**
  - Experiments on the NSL-KDD99 benchmark data using *Linear and Multilayer perceptrons* that match expert knowledge.

- **The advantages:**
  - Applicable to *any classifier* with defined gradients.
  - Does not require modification of the classifier model.
  - Can be extended to further understanding and diagnosis (e.g., vulnerability assessment) of the system.

- **Discussion:**
  - T-SNE does not preserve all information of modified values of initial features (it is a lossy visualization of n-D data)
  - T-SNE converts interpretable features to 2-3 artificial features that have no direct interpretation
  - General Line Coordinates is an alternative to T-SNE that preserve all n-D information, but potentially suffer more from occlusion/clutter in visualization.

---

Do PCA and t-SNE actually **reveal the multi-dimensional relationships?**


The t-SNE author Maaten warned about such statements-- t-SNE may not assign meaning, to point densities, in clusters. The outlier and dense areas, visible in t-SNE, may not be them, in the original n-D space.

In addition, the 2-D attributes, generated by t-SNE, do not have direct **domain interpretation**.

**TensorFlow – visualization tool Embedding Projector**

- a 2D/3D embedding view using principal component analysis (PCA) and t-distributed stochastic neighbor embedding (t-SNE), which “reveal the relationships” between data points with respect to their multi-dimensional representations in a given layer.
- MNIST handwritten images are visualized in t-SNE as rectangles colored by their associated digit labels so that “those images with high similarity in their original feature space are placed close to each other in the 2D/3D space.
- In this manner, “one can easily identify and which digit images are outliers (and thus confusing as another digit)”.

---

Do PCA and t-SNE actually reveal the multi-dimensional relationships?
TreeExplainer for random forests, decision trees, and gradient boosted trees

- The polynomial time algorithm to compute optimal explanations based on game theory.
- An explanation that directly measures local feature interaction effects.
- Tools for understanding global model structure based on combining local explanations of each prediction.
- TreeExplainer matches human intuition across a benchmark of 12 user study scenarios.
- Case Study: three medical machine learning problems where combination of high-quality local explanations reveals global structure while retaining local faithfulness to the original model:
  - trained gradient boosted decision tree
  - local explanations based on SHapley Additive exPlanation (SHAP) values across all samples for understanding global model structure,
  - identified high magnitude but low frequency non-linear mortality risk factors in the general US population,
  - highlighted distinct population sub-groups with shared risk characteristics,
  - identified non-linear interaction effects of risk factors for chronic kidney disease, and
  - monitor a ML model deployed in a hospital by finding features that degrade the model’s performance over time.
- Shapley regression values assign feature importance for linear models.
  - a model is trained with that feature present, and another model is trained with the feature withheld.
  - predictions from the two models are compared on the current input.
  - It requires retraining the model on all feature subsets $S$.
- Shapley sampling values are to explain a model by: (1) applying sampling, and (2) approximating the effect of removing a variable to make it trackable.

**Simple visualization with Local explanations based on TreeExplainer to understand global model structure.**

SHapley Additive explanation (SHAP)

- $f(x)$ is the original prediction model to be explained, $x$ is input to $f$.
- $g(z)$ is the simplified explanation model, $z$ is input to $g$.
- $x'$ is a simplified input to $f$ instead of $x$ through a mapping function $x = h_x(x')$.
- $z'$ is a simplified Boolean input to $g$ instead of $z$ through a mapping function $z = h_x(z')$, $z'=(z'_1, z'_2, ..., z'_M)$ is an Boolean vector of dimension $M$.
- Local methods try to ensure $g(z') \approx f(h_x(z'))$ whenever $z' \approx x'$. (Note that $h_x(x') = x$ even though $x'$ may contain less information than $x$ because $h_x$ is specific to the current input $x$.)
- Definition 1 Additive feature attribution methods have an explanation model that is a linear function of binary variables:

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i,$$

where $M$ is the number of simplified input features, and $\phi_i \in \mathbb{R}$.
- Here $\phi_i$ is to represent the an effect of feature $i$, and summing the effects of all feature attributions approximates the output $f(x)$.
- SHapley Additive explanation (SHAP) assigns each feature an importance value for a particular prediction. It uses conditional expectations to measure the impact of a set of features on the model’s output, averaged over all possible feature orderings.
  - For every possible orderings, features are introduces one at a time into a conditional expectation of the model’s output.

Activation patterns of individual hidden nodes: LSTMVis system

- LSTMVis [Kahng et al. 2018]: Interactive exploration of the learnt behavior of hidden nodes
- A user selects a phrase, e.g., "a little prince," and specifies a threshold the system
  - shows hidden nodes with activation values greater than the threshold and
  - finds other phrases for which the same hidden nodes are highly activated.
  - Given a phrase in a document, the line graphs in the top panel visualize the activation patterns of hidden nodes over the phrase.
- Several other works with a similar idea – activation and heatmap.
- Open questions:
  - In the nearest neighbor explanation is by the most similar case. Here it does not explain why the activation makes sense.
  - Where are relations between salient element is captured in this visualization?
  - How to measure that the explanation is right?
- Visual tools are limited by Heatmap and Parallel coordinates


Approaches to **discover analytical** ML models boosted by visual means

- Discovering visual ML models assisted by analytical ML algorithms, such as
  - propositional and first order rules,
  - random forests,
  - CNN,
  - decision trees,
  - optimization based on genetic and other algorithms
Quality-based guidance for exploratory dimensionality reduction

- Problem:
  - Difficulty or inability to represent effectively high-dimensional data with hundreds of variables by visualization methods.

- Known solutions:
  - Employing dimensionality reduction (DR) prior to visualization.

- Challenge:
  - DR can throw the baby with bathwater – loss of information of full high-dimensional data.

- Approach: interactive environment to understand high-D data first with:
  - Discovering structure of full high-dimensional data without reduction.
  - Identifying importance and interestingness of structures and variables.
  - Using several metrics of ‘interestingness’ of variables.
  - Using PCA to combine metrics and get new “principal” metrics.

- Use case:
  - DNA sequence-based study of bacterial populations.

Quality-based guidance for exploratory dimensionality reduction

Top - visualizations of quality metrics of variables using parallel coordinates and glyphs.
In parallel coordinates,
axes are quality metrics, and
polylines are the variables.
In the glyph plot,
axes are first two “principal metrics”
glyphs are located at the points with values of these metrics for each variable.
glyph is a set of squares
each square represents one metric
opacity of the square represents the metric value.
the fill color of the square is the same as the axis of the parallel coordinates
the border color of the glyph is the color of polylines in parallel coordinates.

Highest ranked operational taxonomic units (OTUs) in a scatter plot matrix, where positive and negative correlations are represented by blue and red cells.

The variable merging window, including a list of suggestions for variable groups to merge.
An Interactive Large-Scale Graph Mining and Visualization

Problem:
- Exploring efficiently a large graph with several millions or billions of nodes and edges, such as a social network.

Solution:
- Perseus, an integrative system for analysis of large graphs

Approach:
- summarization of graph properties and structures,
- guiding attention to outliers,
- exploration of normal and anomalous node behaviors.
- automatic extraction of graph invariants (e.g., degree, PageRank, real eigenvectors)
- scalable online batch processing on Hadoop;
- visualization of 1-D and 2-D distributions of invariants
- Visualization of a subgraph of the selected node and its neighbors, by incrementally revealing its neighbors.

Use Cases: multi-million-edge social networks
- Wikipedia vote network,
- friendship network in Slashdot, and
- trust network based on the consumer review website Epinions.com.

Contrastive and Visual Topic Modeling for Comparing Documents

Problem:
- How to express similarities/differences of ‘labeled’ document collections and visualize them efficiently?

Approach
- Learn hidden topics and embeddings for the documents, and labels for visualization.
- Extract hidden discriminative and common topics across labeled documents.
- Visualize all documents, labels, and the extracted topics, where proximity in the coordinate space is reflective of proximity in semantic space;
- Extract topics and visual coordinates simultaneously under a joint model.
- Probabilistic approach to create a visualization space that differs from t-SNE.

Results:
- Outperforms both unsupervised and supervised state-of-the-art topic models in contrastive power, semantic coherence and visual effectiveness on real world data.
- Interpretation?

Contrastive analysis of sports, arts, style documents from NYTnews (20 topics).
Contravis: (supervised joint method) finds latent discriminative topics and gives visual embedding of documents and topics. Two discriminative topics per label shown in wordclouds (w/ respective colors) + one common topic (black);

Monotone Boolean Function visualization vs. Parallel Coordinates for binary data

(c) cancer visualization in centered chain order

(f) biopsy visualization in centered chain order

(j) Highly overlapped parallel coordinate visualization of the same data (yellow - benign, red –malignant)

(k) Types of source X-ray mammography images used producing Boolean vectors

Boolean Data and Hansel Chains

- Boolean Data are data composed of ‘0’ and ‘1’
- Hansel Chains browse without the binary cube or hyper cube without overlapping

Traditional graph representation uses space inefficiently (unused space)
The concept of the monotone Boolean function. Let $E^n = \{0, 1\}^n$ be a binary $n$-dimensional cube then vector $y = (y_1, y_2, \ldots, y_n)$ is no greater than vector $x = (x_1, x_2, \ldots, x_n)$ from $E^n$ if for every $i$, $x_i \geq y_i$, i.e.,

$$x \geq y \iff \forall i \ x_i \geq y_i.$$ 

In other words, vectors $x$ and $y$ are ordered. In general relation $\geq$ for Boolean vectors in $E^n$ is a partial order that makes $E^n$ a lattice with a max element $(1, 1, \ldots, 1)$ and min element $(0, 0, \ldots, 0)$. Boolean function $f: E^n \to E$ is called a monotone Boolean function if

$$\forall x \geq y \Rightarrow f(x) \geq f(y).$$

This monotonicity property implies two expansion properties for function $f$:

$$x \geq y \ & f(y) = 1 \implies f(x) = 1, \quad x \geq y \ & f(x) = 0 \implies f(y) = 0,$$

**Space structure**

**Data in the space structure**

This MONOTONICITY property implies two expansion properties for function $f$:
Lossless Visualization of 48-D and 96-D data in CPC-Stars, Radial Stars and Parallel Coordinates

Examples of corresponding figures: stars (row 1) and PCs lines (row 2) for five 48-D points from two tubes with $m = 5\%$. Row 3 and 4 are the same for dimension $n=96$.

Two stars with identical shape fragments on intervals [a,b] and [d,c] of coordinates.

Samples of some class features on Stars for n=48.

Samples of some class features on PCs for n=48.

Visual Patterns-- combinations of attributes
Human abilities to discover high-dimensional interpretable patterns in 160-D

Twenty 160-D points of 2 classes represented in star CPC with noise 10% of max value of normalized coordinates (max=1) and with standard deviation 20% of each normalized coordinate.

(a) Initial 100-D points without noise for Class (Hyper-tube) #1 and Class (Hyper-tube) #2

(b) 100-D points with multiplicative noise: circled areas are the same as in upper star.

Figure 6.10. Samples of 100-D data in Star CPC used to make participants familiar with the task.

Twenty 160-D points of 2 classes represented in Radial Coordinates with noise 10% of max value of normalized coordinates (max=1) and with standard deviation 20% of each normalized coordinate.

(a) Initial 100-D points without noise for Class (Hyper-tube) #1 and Class (Hyper-tube) #2

Figure 6.10. Samples of 100-D data in Star CPC used to make participants familiar with the task.

Twenty 160-D points of 2 classes represented in Parallel Coordinates with noise 10% of max value of normalized coordinates (max=1) and with standard deviation 20% of each normalized coordinate.
The experiment with $n=192$ and a high level of noise (30%) points out on the likely upper bound of human classification of n-D data using the Radial Coordinates for data modeled as linear hyper-tubes.

The experiment with $n=160$ shows that the upper bound for human classification on such n-D data is no less than $n=160$ dimensions with up to 20% noise.

Thus the expected classifiable dimensions are in $[160,192]$ dimensions interval for the Radial Coordinates.

Due to advantages of Star CPC over Radial Coordinates, these limits must be higher for Star CPC and lower for Parallel Coordinates due to higher occlusion in PC.

More exact limits are the subject of the future experiments. About 70 respondents participated in the experiment with 160-D, therefore it seems that 160 dimensions can be viewed as a quite firm bound.

In contrast, the question that 192-D is the max of the upper limit for Star CPC may need addition studies.

Thus, so far the indications are that the upper limit for Star CPC is above $n=192$ and it needs to be found in future experiments for linear hyper-tubes. Finding bounds for linear-hyper-tubes most likely will be also limits for non-linear hyper-tubes due to their higher complexity.
Heatmap for non-image data

- This case study uses the WBC data with the Collocated Paired Coordinates (CPC-R) algorithm, for converting non-image data to images, and CNN algorithms for discovering the classification model in these images.
  - Each image represents a single WBC data case, as a set of squares with a different level of intensities and colors.

- The CPC-R algorithm is a modification of Collocated Paired Coordinates (CPC) algorithm.
  - The CPC algorithm first splits attributes of an n-D point $x=(x_1, x_2, \ldots, x_n)$ to consecutive pairs $(x_1, x_2), (x_3, x_4), \ldots (x_{n-1}, x_n)$. If $n$ is an odd number then the last attribute is repeated to get $n+1$ attributes. Then all pairs are shown as 2-D points in the same 2-D Cartesian coordinates and connected by arrows to form a directed graph $x^*$: $(x_1, x_2) \rightarrow (x_3, x_4) \rightarrow \ldots \rightarrow (x_{n-1}, x_n)$. This graph is equivalent to the n-D point $x$ and it can be fully restored from the graph.

![](image.png)

6-D point (5,4,0,6,4,10) in Collocated Paired Coordinates.
The CPC-R algorithm, instead of connecting pairs \((x_1, x_2)\) by arrows, uses the grey scale intensity from black for \((x_1, x_2)\) and very light grey for \((x_{n-1}, x_n)\) for cells. Alternatively, intensity of a color is used. This order of intensities allows full restoration of the order of the pairs from the image.

The size of the cells can be varied from a single pixel to dozens of pixels.
- E.g., if each attribute has 10 different values then a small image with 10x10 pixels can represent 10-D point by locating five grey scale pixels in this image.
- This visualization is lossless when values of all pairs \((x_i, x_{i+1})\) are different and do not repeat. An algorithm for treatment of colliding pairs is presented in [21].

Figure (a) shows the basic CPC-R image design and Figure (b) shows a more complex design of images, where a colored CPC-R visualization of a case is superimposed with mean images of the two classes, which are put side by side, creating double images.

The experiments with such images reached accuracy over 97.30% in 10-fold cross-validation for different CNN architectures on WBC data [Kovalerchuk, Agrawal, 2019].

The advantage of CPC-R is in lossless visualization of n-D cases, and the ability to overlay them using heatmap with salient points discovered by the CNN model, for model explanation.

- 6-D point \((5,4,0,6,4,10)\) in Collocated Paired Coordinates.

(a) 10-D point \((8, 10, 10, 8, 7,10, 9,7,1,1)\) in CPC-R. (b) Visualization in colored CPC-R of a case superimposed with mean images of two classes put side by side. CPC-R visualization of non-image 10-D points.
Conclusion

- The tutorial covered four complementary approaches:
  1. to **visualize** ML models produced by **analytical** ML methods,
  2. to **discover** analytical ML models boosted by **visual** means,
  3. to **explain** deep and other ML models by visual means,
  4. to **discover visual** ML models boosted assisted by **analytical** ML algorithms,

- All of them benefit data science now and will continue in the future.
New tasks for visual knowledge discovery

- Expansion to **new applications** in finance, medicine, NLP, image processing and others
- Deepen **links** with Deep Learning methods
- Expansion of hybrid approach to **prevent overgeneralization** and overfitting of predictive models by using visual means.
- **Multiple “n-D glasses”** as a way to super-intelligence and virtual data scientists.
Future of Interpretability and XAI

• Creating simplified explainable models with prediction that humans can actually understand.
• “Downgrading” complex Deep Learning models for humans to understand them.
• Expanding visual and hybrid explanation models.
• Further developing explainable Graph Models.
• Further developing ML model in First Order Logic (FOL) terms of the domain ontology.
• Generating models with the sole purpose of explanation.
• Post-training rule-extraction.
• Expert-in-the-loop in the training and testing stages with auditing models to check generalizability of models to wider real-world data.
• Rich semantic labeling of a model’s features that users can understand.
• Estimating the causal impact of a given feature on model prediction accuracy.
• Using new techniques such as counter-factual probes, generalized additive models, generative adversarial network technique for explanations.
• Further developing heatmap visual explanations of CNN by Gradient-weighted Class Activation Mapping and other methods with highlighting the salient image areas.
• Adding explainability to DL architectures by layer-wise specificity of the targets at each layer.
REFERENCES


Kovalerchuk B., Grishin V. Adjustable General Line Coordinates for Visual Knowledge Discovery in n-D data, Information Visualization, 18(1), 2019, pp. 3-32.

REFERENCES

Kovalerchuk B., Grishin V. Adjustable General Line Coordinates for Visual Knowledge Discovery in n-D data, Information Visualization, 18(1), 2019, pp. 3-32.


