# Extracting Knowledge with Constructivist Machine Learning: Conceptual and Procedural Models

Thomas Schmid[a,b], Florian Grosse[b]

[a]Lancaster University in Leipzig, Germany
[b]Universität Leipzig, Germany

**Abstract**

As the most essential knowledge engineering technique, the process of knowledge extraction has been widely researched and applied in many domains of computer science. While other knowledge engineering methods have lately received increasing attention in combination with machine learning, the extraction of structured knowledge is rarely addressed with other than standard machine learning tasks like classification. To this end, we have introduced a novel approach for hybrid artificial intelligence based on constructivist learning theories. Termed Constructivist Machine Learning (conML), our frameworks provides improved interpretability by utilizing metadata for the creation and management of a data-driven knowledge base. Here, we explain and demonstrate how the *conML* framework may be employed to extract procedural or conceptual knowledge from data where a time stamp, sensor ID and specific purpose are available as metadata for each data sample. As an illustrative example, we extract both conceptual and procedural models from data modelled after spectroscopic measurements. For the resulting procedural and conceptual knowledge bases, we observe an automated generation and adaption of models with explicitly defined validity and ranging over up to three levels of abstraction. From this, we conclude that a constructivist knowledge base provides valuable insights into a given data set.

**Keywords**

Hybrid Artificial Intelligence, Knowledge Extraction, Knowledge Discovery, Constructivist Machine Learning

## 1. Introduction

Originally invented to process numerical data, computers have developed into an essential tool for today's information industries and societies. Transferring human-understandable information into processable semantic information has therefore become a classic task in modern computer science. This task, commonly referred to as knowledge extraction, is traditionally associated with either structured data or the field of natural language processing. Oftentimes, the term knowledge extraction is used to summarize tasks such as named-entity recognition or the generation of an ontology. A common result of such processes is a set of entities and relations, which is referred to as a knowledge base and allows for further automated processing, e.g., with methods of artificial intelligence (AI).

In fact, knowledge bases in this sense are an essential foundation of widespread applications such as search engines or chat bots today. Knowledge bases employed in commercial products and services, however, are often not readily transparent to users. Among a relatively small number of large and widely recognized public knowledge bases are Wikidata [1, 2], DBpedia [3] and FreeBase [4]. Due to the increasing complexity of available data and information, rule-based approaches have been proposed to acquire such structured knowledge automatically (cf. [5, 6]). Well-known examples for

*In A. Martin, K. Hinkelmann, H.-G. Fill, A. Gerber, D. Lenat, R. Stolle, F. van Harmelen (Eds.), Proceedings of the AAAI Spring Symposium on Machine Learning and Knowledge Engineering for Hybrid Intelligence (AAAI-MAKE 2022) - Stanford University, Palo Alto, California, USA, March 21-23, 2022.*

schmid@informatik.uni-leipzig.de (T. Schmid)

knowledge extraction systems are, e.g., YAGO [7, 8], OpenIE [9] and Knowledge Vault [10].

In recent years, interest has grown in employing machine learning techniques for knowledge extraction and knowledge engineering [11], e.g., for the automatic or semi-automatic construction of ontologies by so-called ontology learning [12]. Growing research activities in this area of hybrid AI are largely motivated by the complementary strengths of machine learning and knowledge engineering [13]: Machine learning is able to process real-world data like sensory inputs and situations in which no explicit knowledge is available or knowledge is tacit. Explicitly represented knowledge, on the other hand, generally provides better interpretability of results and behaviour, but requires higher initial effort for collecting, encoding and exploiting the knowledge.

Addressing these challenges and goals, we have introduced an approach termed Constructivist Machine Learning [14] which is based on the influential philosophical concept of constructivism [15]. Constructivism describes the idea that humans actively construct or create their own knowledge, and that reality is determined their individual experiences as a learning subject [16]. Adapting this multi-faced view of the world, our framework allows to identify meaningful blocks of data and propose models from them, relate them to each other in a hierarchical scheme similar to a traditional knowledge base and update them when necessary [17]. In particular, this system is designed to both recognize ambiguity and avoid it in its knowledge base through active selection and evaluation of data and models [18]. Here, we describe a prototypical application in which our framework is used to extract two distinct types of knowledge from a given data set. We employ a synthetic data set modeled after spectroscopy measurements on epithelial tissue [14], illustrate the constructivist knowledge extraction process for this data and review the resulting metadata-based hierarchical knowledge base.

## 2. Constructivist Concepts of Knowledge

The question of what constitutes knowledge has been answered in the past by different disciplines, sometimes very differently. In artificial intelligence, e.g., a prominent definition describes knowledge as ”whatever can be ascribed to an agent, such that its behavior can be computed according the principle of rationality” [19]. In fact, a whole subcommunity of AI researchers has evolved with the shared goal of formalizing and representing knowledge in a processable way [20]. Not surprisingly, however, the rather fundamental question for the nature of knowledge has been raised and addressed by other scientific disciplines, too (e.g., [21, 22, 23, 24, 25]).

While the knowledge engineering community has developed a rich spectrum of techniques to represent knowledge (cf. [26]), it has to be stated that these approaches were originally not intended to be included in an adaptive process like machine learning. Moreover, traditional knowledge representation concepts are grounded in the complementary ideas of belief and ”truth”, both of which are are at odds with modern philosophical concepts such as constructivism. Constructivism has not only questioned classical ideas of truth and facticity [27], but moreover argued for time- and subject-boundedness of any knowledge construct[1]. Such boundaries are, however, often not an explicit component of standard knowledge representations.

Therefore, we will use a concept of knowledge based on constructivist theories of learning outcomes here, which is the rationale behind most current eductional concepts in schools and higher education [30, 31]. In this context, learning constitutes either from constructing, reconstruction or deconstructing knowledge [28], which forms the basis for our Constructivist Machine Learning framework [15]. Such constructivist teaching designs also assume a hierarchical ordering of learning goals,

[1] cf. [28]: ”In the assertion of the construction of reality is included that such constructions are in each case time-bound, depend on the specific observers and their community of understanding, that they cannot establish eternal truths [...].”

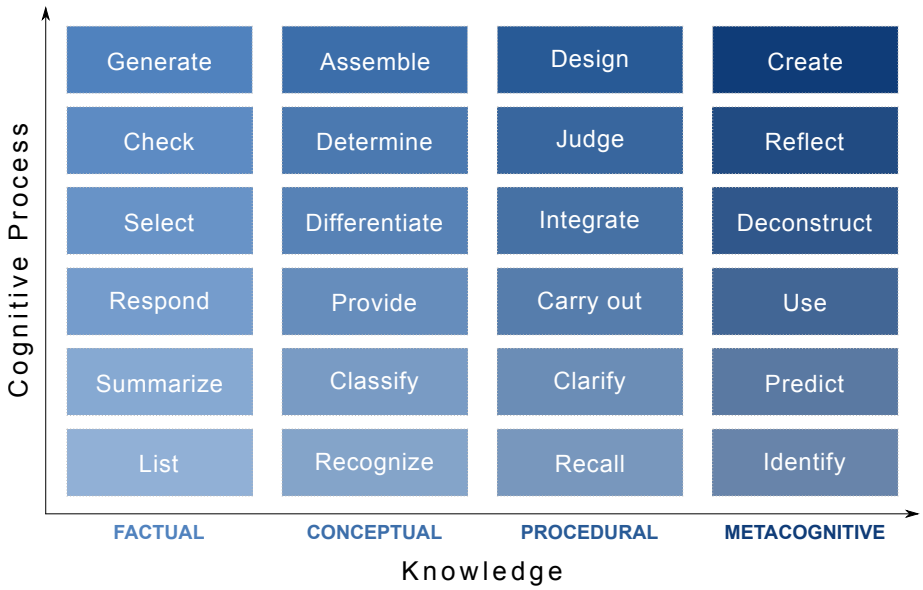



**Figure 1:** A two-dimensional representation of Bloom's Revised Taxonomy of Educational Objectives [29]. The knowledge dimension is representated on the x-axis, the cognitive process dimension on the y-axis.

which is adapted here. To this end, Bloom's taxonomy for cognitive learning goals [32] is particularly well known. After an interdisciplinary scientific debate [33, 34, 35], a revised version was proposed [29] that still represents a state-of-the-art description of human cognitive skills and an important working basis in practical pedagogy and didactics. The core of Bloom's taxonomy is the hierarchical order of cognitive processes in six levels [29]. The lowest level corresponds to pure remembering or recognizing. The next two higher levels are first the ability to understand or to recognize meaning, and then the application of a learned process in a specific situation. As higher cognitive processes then follow the analysis and evaluation of objects and processes. The highest level is reached with a synthesis, i.e. the creation of a new product.

In a second dimension, called the knowledge dimension, Bloom's revised taxonomy considers four distinct knowledge domains in which different cognitive processes take place and different types of knowledge is acquired [29]: a factual, a conceptual, a procedural and a metacognitive knowledge domain (see also Fig. 1). Factual knowledge in this sense refers to knowledge of terminology or specific details and elements [36], and is to some extend the type of knowledge represented by ontologies. Conceptual knowledge includes classifications and categories, principles and generalizations as well as theories, models and structures [36]. Procedural knowledge, on the other hand, comprises subject-related skills or algorithms, subject-related techniques and methods as well as criteria for the selection of suitable methods [36]. Finally, metacognitive knowledge refers to strategic knowledge, knowledge about cognitive tasks (e.g., appropriate context and conditions) or self-knowledge [36].

As motivated in our previous work [15], we assume that metadata are an essential prerequisite to learn, adapt and hierarchically organize models [17]. All learned models in the context of Constructivist Machine Learning are considered pragmatic models in the sense of Stachowiak's General Model Theory [37], as this concept employs built-in metadata in form of pragmatic properties. Such Stachowiak models allow not only to represent mathematical functions by means of machine learning, but incorporate metadata about the validity of the model regarding subject, purpose and time:

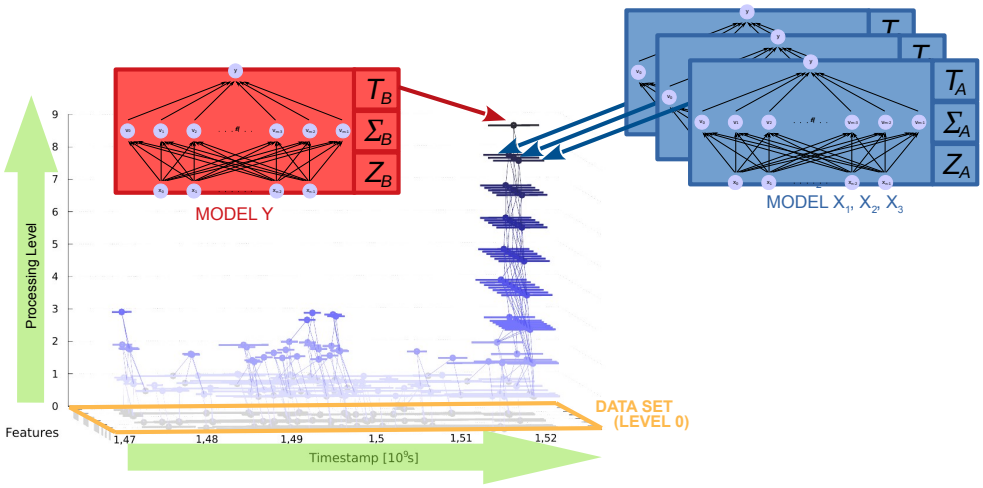



**Figure 2:** Example of a Constructivist Knowledge Base. Three-dimensional representation of a hierarchically ordered set of models created by Constructivist Machine Learning for a given knowledge domain. A top-level model (red) is highlighted, whose output $Y$ depends on the outputs $X_1$, $X_2$ and $X_3$ of three models located on a lower level (blue); for these four depicted models, a neural network model is used here. The processing level dimension in inspired by, but not identical with Bloom's Revised Taxonomy of Educational Objectives. The original input data is represented on processing level 0.

- The author, user or *subject* $\sigma$, of a model may be a sensor or a measuring device in natural sciences, or typically a human evaluator in observational studies or content analyses. The set of all model subjects $\sigma_i$, for which a given model $\mathcal{M}$ is valid, is called $\Sigma_\mathcal{M}$ and defined as the subset of the (infinite) set $\Sigma$ of all possible subjects ($\Sigma_\mathcal{M} \subset \Sigma$)
- The target parameter of a model $\mathcal{M}$ is referred to as *purpose* $\zeta$ and reflects what is to be achieved by $\mathcal{M}$. The set of all purposes $\zeta_i$, for which a given model $\mathcal{M}$ is valid, is called $Z_\mathcal{M}$ and defined as subset of the (infinite) set $Z$ of all possible model purposes ($Z_\mathcal{M} \subset Z$).
- The *temporal validity* of a model $\mathcal{M}$ may be represented by a time span $T_\mathcal{M}$ within which $\mathcal{M}$ is applicable (and outside which it is not). $T_\mathcal{M}$ is described using an interval defined by a lower boundary $\tau_{min}$ and an upper boundary $\tau_{max}$, i.e., $T_\mathcal{M} = [\tau_{min}, \tau_{max}]$.

With $T_\mathcal{M}$, $\Sigma_\mathcal{M}$ and $Z_\mathcal{M}$ given for each learned model, models learned by Constructivist Machine Learning may be related to each other and displayed in a hierarchical knowledge base (Fig. 2).

## 3. Constructing Conceptual and Procedural Knowledge Bases

The overall motivation of applying Constructivist Machine Learning is to identify and organize machine learning models not only based on data for training and testing, but also on metadata [17]. Incorporating them allows to deconstruct such models by abstraction, differentiation or discarding wherever related models can be identified [18]. Moreover, we have proposed that a hierarchically ordered set of models created in this way constitutes an enriched knowledge base [15]. Following the fundamental concepts of Bloom's Revised Taxonomy of Educational Objectives, we distinguish four different types of knowledge (cf. Fig. 1). In the present work, our focus lays on conceptual and procedural knowledge as machine learning technique appropriate to create representations for such knowledge may be identified relatively easily for both of these two cases.

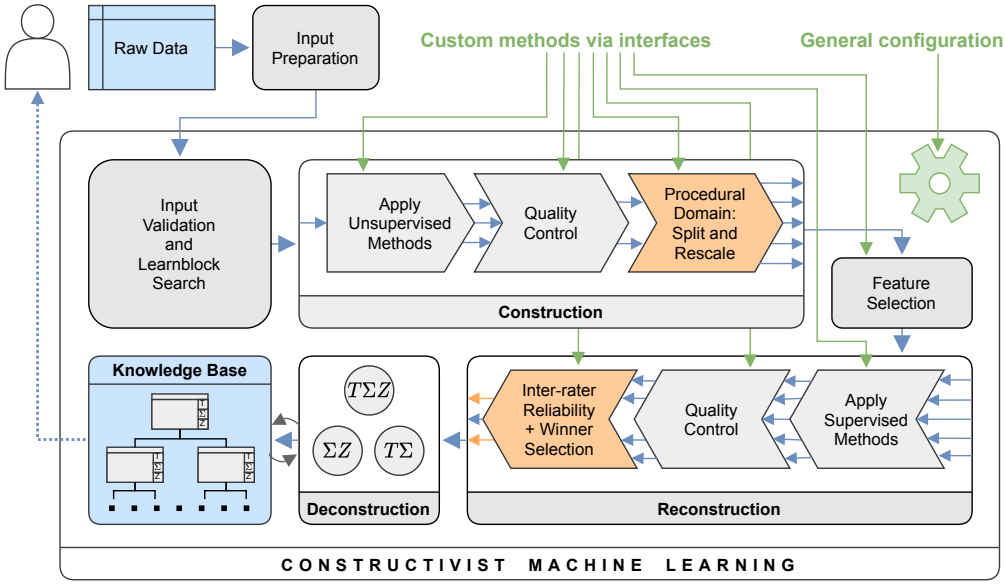



**Figure 3:** Overview of the Constructivist Machine Learning framework. Construction, Reconstruction, and Deconstruction refer to the key learning processes [17, 18]. Blue arrows indicate the flow of data during the overall process. Green arrows denotes a user's possibilities to influence the overall process through custom configurations. In orange color, components are highlighted in which algorithmic processes differ between procedural and conceptual learning. (Figure adapted from [38])

The overall process of Constructivist Machine Learning consisting of construction, reconstrucion and deconstruction has been proposed in previous work [17]. For the construction process, the algorithmic procedures proposed by us follow an unsupervised learning paradigm; for the reconstruction, a supervised learning strategy is employed. For the deconstruction, we have created a novel paradigm, which we have described in more detail in a follow-up work [18]. In parallel, we have published Python, R and Julia source code for this learning framework[2]. Here, we will therefore limit our description of the constructivist learning to algorithmic differences between generating a conceptual knowledge base on the one hand and a procedural knowledge base on the other hand.

In terms of Constructivist Machine Learning, the difference between these two types of knowledge is most apparent during the model construction process. In an educational context, construction is generally associated with creativity, innovation and production, and in particular with the search for new variations, combinations or transfers [39, p. 145]. For constructing machine learning models, we interpret this as an unsupervised learning process that identifies or defines alternative $n$-dimensional outputs to a set of $m$-dimensional outputs [15, 17]; thus, it is assumed that target values are either not known or not considered. Consequently, the construction step aims to identify as many different (valid) machine learning models as possible and to thereby create competing model candidates, which will be evaluated in a following reconstruction process. Rationale behind this is that it is a priori unclear which of the models constructed for a given set or subset of data samples might be best choices regarding accuracy and intersubjectivity. To select the most appropriate model for a given set or subset of data, these competing models are evaluated by means of supervised learning during the following reconstruction process.

[2] Available for download via www.constructivist.ml/download

### 3.1. Conceptual Models

Conceptual knowledge in the sense of Bloom's Revised Taxonomy of Educational Objectives may be described as knowledge about the interrelationships "among the basic elements within a larger structure" [36], in particular about classifications and categories. It may further comprise knowledge about principles and generalizations, as well as about theories, models and structures [36]. Here, we outline how such knowledge may be extracted in the context of Constructivist Machine Learning.

**Reconstruction.** Algorithms that learn distinguishable categories within a given data set are commonly summarized as classification models or classifiers. In order to learn such classes or categories, however, they need to be known before the actual learning takes place. Where this is the case, Constructivist Machine Learning provides a process called reconstruction, which is intended to adapt existing class mappings for given data. As classification algorithms, our framework currently supports either algorithms provided by the Scikit-Learn library (for the Python version) or that implement a Scikit-Learn API (for the Julia version); or algorithms of the CRAN repository (for the R version).

**Winner Selection.** If a reconstructed model passes user-defined accuracy and intersubjectivity criteria, it is considered for the selection of a winner model, which is expected to be the optimal representation of the data block. In Constructivist Machine Learning, this step is handled within the reconstruction process. To this end, models are ranked in descending order using Krippendorff's $\alpha$, where a maximum $\alpha$ value implies the maximum degree of intersubjectivity. If two or more models demonstrate the same $\alpha$ value, the model with the smallest feature space is preferred in this subset.

**Construction.** More commonly, however, neither number of classes or categories nor appropriate labels for them can be known. Rather, such knowledge needs to be created from the ground up. For situations where this is the case, Constructivist Machine Learning provides the so-called construction process, which is intended to identify an optimal mapping of classes or categories for given data. In an unsupervised learning fashion, such mappings may be identified by employing clustering algorithms.

The use of clustering methods usually entails a number of parameter choices to be made. Most commonly, the expected number of clusters needs to be specified, but algorithms that automatically determine an appropriate number of clusters are also available. Often, the same algorithm is used to perform several runs with different hyperparameters and evaluate the obtained clusterings with an external procedure [40]. Here, the maximum cluster count (for algorithms that allow direct specification of such) is referred to as the maximum categorical complexity $\kappa_k$. Each such clustering procedure thus generates $\kappa_k - 1$ models with $k = \{2, ..., \kappa_k\}$ clusters or categories. Algorithms that determine the number of clusters by other means may be executed with a set of possible configurations instead.

Note that not all constructed conceptual models will be considered for integration into the knowledge base (cf. Fig. 2). Some models may be discarded during the construction process, and all models leaving the construction process will be evaluated by undergoing a consequent reconstruction process (cf. Fig. 3). As mentioned, all models entering the reconstruction process assessed regarding accuracy as well as regarding intersubjectivity, and at maximum one conceptual model at a time may be considered for integration into the knowledge based.

### 3.2. Procedural Models

Procedural knowledge in the sense of Bloom's taxonomy may be described as knowledge on "how to do something" [36], in particular knowledge of subject-specific skills and algorithms. It may further comprise knowledge of subject-specific techniques and methods and knowledge of criteria for determining when to use appropriate procedures [36]. Here, we outline how such knowledge may be extracted in the context of Constructivist Machine Learning.

**Reconstruction.** In the following, procedural knowledge is therefore interpreted as the total set of machine learning models implementing multilateral regression tasks. In order to learn one or more regressions, however, continuous target values need to be known before the actual learning takes places. For situations where this is the case, our framework provides the reconstruction process intended to relate existing continuous target values to input values for given data. As regression algorithms, our framework currently supports either the algorithms provided by the Scikit-Learn library (for the Python versionof Constructivist Machine Learning) or all algorithms that implement a Scikit-Learn API (for the Julia version) or provided by the CRAN repository (for the R version)

**Scaling.** While differences in scale are negligible in the classification tasks of the conceptual knowledge domain, the creation of new metric targets via multiple unsupervised models in the procedural domain may lead to features with mismatching value ranges where models use the output of other models as inputs. Therefore, we have proposed the an additional normalization step for the targets [38]. This ensures that the targets for all candidate models lay in a similar range and makes this range independent of the specific unsupervised method employed and of the input data. In particular, this allows for the targets of models on lower knowledge levels to become features for models on a higher level. Thus, the target rescaling step acts as an input normalization for higher-level model candidates.

**Winner Selection.** In contrast to the conceptual domain, a single candidate in the procedural domain – representing a single artificial metric feature – is typically not characteristic for all datapoints of the model that created it. Unsupervised methods that yield metric features, such as manifold learning, feature learning or other dimensionality reduction techniques, tend to create models of their input spaces that distribute information over all generated output features. A single output feature therefore typically contains less global information about the input space than a comparable classification. The globality or locality of the information in the features may vary with the specific method, but a single output feature can only be descriptive for global input space of minimal complexity.

Thus, the goal here is to select a subset of those models that allows to extract and describe as much knowledge as possible about the input space while removing redundant and meaningless information. The key insight is that this problem actually corresponds very directly to an unsupervised feature selection task. Therefore, we have reformulated the winner selection problem for the procedural domain as an unsupervised feature selection task and implemented a selection scheme based on an extension of the broadly applicable minimal-redundancy-maximal-relevance framework (mRMR) for unsupervised feature selection [38]. Originally proposed by Peng, Long and Ding [41], this technique allow to control redundancy by leveraging the known interrater reliability scores and correlation coefficients between candidates, but no explicit target values.

**Construction.** Unlike regular regression, no information about continuous target values is available when creating procedural knowledge from the ground up. Rather, such values have to derived by unsupervised learning techniques like dimensionality reduction. The construction of new procedural models is therefore implemented here as the extraction of previously unknown metric features, such that every output dimension of either algorithm is regarded as a separate model candidate.

## 4. Case Study: Knowledge Extraction from Spectral Data

To illustrate the process of extracting either a conceptual or a procedural knowledge base, we performed an exemplary case study using the Julia implementation of our framework[3] on synthesized spectroscopy measurements. This data is the result of our previous research on modeling impedance measurements; motivation and rationale for this have been described in more detail elsewhere [14].

[3] Available for download via www.constructivist.ml/download

## 4.1. Data

We used a data set of 275,000 samples with 348 features and three features considered as metadata for each sample. The 348 features are composed of different representations of impedances obtained at 42 frequencies between 1.3 and 16,350 Hz. An impedance is a physical measurand that generalizes the principle of an electrical resistance beyond DC circuits and classical ohmic resistance. Originally, these 42 impedance are displayed as complex values in cartesian coordinates, but are additionally transformed into alternative representations like magnitude and phase. Those different representations are used to delineate feature subsets that form the input for the following knowledge extraction steps. The three metadata features represent temporal, subjective and purpose-related limits required for the generation of Stachowiak-like models.

Semantically, these samples mimick impedance spectroscopic measurements of three different types of epithelial tissue under four different functional states. Epithelia are one of four basic tissue types of the human body and form single- or multi-layered tissues covering many internal and external body surfaces. The three types considered here originate in the human colon, the pig jejenum and the kidneys of dogs. The functional states are related either to control conditions or the application of one or more drugs. For more details on tissues and properties see also [14].

For both conceptual and procedural knowledge base generation, the feature subsets of the data set are analyzed independently one after another in combination with the respective metadata. Each input subset is then further divided into temporally contiguous blocks of fixed size that are processed consecutively and individually by the Constructivist Machine Learning pipeline.

## 4.2. Conceptual Knowledge

In a first experiment, a constructivist knowledge base consisting only of conceptual models is created for the given data. While the same subset splitting of the data is used for both conceptual and procedural knowledge extraction, the employed algorithms as well as the employed parameters differ. Table 1 gives an overview of the applied parameter settings.

**Learning Algorithms.** We use three alternative clustering methods for the construction process:

1. KMeans[4]
2. Hierarchical Clustering with Ward linkage[5]
3. Hierarchical Clustering with complete linkage and Manhattan metric[6]

Both KMeans and Hierarchical Clustering are statistically motivated clustering algorithms. But while KMeans is computationally efficient but unstable to random value initialization, hierarchical clustering is computationally costly but provides a well interpretable tree of alternative clusterings.

For the reconstruction process, we use two distinct supervised techniques from the areas of biologically inspired and statistically motivated learning:

1. Multi-Layer Perceptron[7] (number of input neuron depending on model size, one hidden layer with number of hidden neurons depending on model size, tanh activation function for hidden neurons, training with lbfgs, 3-fold cross-validation)
2. AdaBoost[8] (number of decision stumps: 10)

[4]Implemented using https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
[5]Implemented using https://scikit-learn.org/stable/modules/generated/sklearn.cluster.ward_tree.html
[6]Implemented using https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html
[7]Implemented using https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html
[8]Implemented using https://github.com/bensadeghi/DecisionTree.jl

| | Parameter | Setting |
|---|---|---|
| Batch Processing | UseLearnBlocks | biggest |
| | LearnBlockMinimum | 2000 |
| Construction | MinCategorySize | 0.1 |
| Complexity Reduction | MaxFeatures | 10 |
| | maxFilterFeatures | 50 |
| | maxFilterSamples | 15000 |
| | MaxModelReduction | true |
| Reconstruction | testSetPercentage | 0.33 |
| | MinTestAccuracy | 0.8 |
| | KrippendorffMetric | nominal |
| | MinReliability | 0.8 |
| | AllowWeakReliability | true |

| | Parameter | Setting |
|---|---|---|
| Deconstruction | deconstructionStrategy | integrative |
| | deconstructionMode | maximal |
| | SigmaZetaCutoff | 0.2 |
| | MaxDistanceT | 1.0 |
| | FullTolerance | 0.1 |
| | TΣTolerance | 0.0 |
| | MaxTimeConflicts | 0.2 |
| Other | highestLevel | 9 |
| | usePredictionsAsTargets | false |
| | enablepile | true |
| | enablewaste | false |
| | lscvBandwidth | true |
| | learningDomain | Conceptual |

**Table 1**
Settings for Extracting Conceptual Knowledge with Constructivist Machine Learning. Parameters are grouped according to the step in the pipeline that is influenced. See [14] for detailed explanation.

These methods represent a prominent biologically motivated learning algorithm (neural networks) on the one hand, and a modern statistical ensemble learning method (AdaBoost) on the other. Also, both methods are flexible enough to be employed in regression tasks analogously.

**Training Progress.** Figure 4 gives an overview of the progress during the knowledge extraction process. At the beginning of training, a large number of models is added to the first level of the knowledge base (illustrated by Fig. 4b). As soon as alternative representations of the same data are introduced (fourth feature subset and following), the learning by deconstruction sets in which results in the deletion of conflicting models and the appearance of higher-level models that represent abstractions of the models that they derive from (illustrated by Fig. 4c). The total number of stored models seems to saturate over the second half of the training procedure, with some of the abstracted models being removed again, possibly by model fusion or arising conflicts with new data. Whether this is due to the inherent limit in complexity in the dataset or suboptimal choices of algorithms and/or hyperparameters has not been investigated in detail yet.

**Results.** After all 18 batches have been processed, a constructivist knowledge base has been generated with 53 conceptual models on processing level 1 and 14 conceptual models on processing level 2. This implies that some previously integrated models have been removed (e.g., from processing level 3) by means of deconstruction during the training process (cf. 4a), which is intended [18].

## 4.3. Procedural Knowledge

In a second experiment, a knowledge base consisting only of procedural models is created with the same input data. The overall settings are summarized in Table 2.

**Learning Algorithms.** For the construction process, we use two distinct unsupervised techniques from the areas of biologically inspired and statistically motivated multivariate regression:

1. Autoencoder[9] (one hidden layer with a user-defined number of hidden neurons, sigmoid activation function for hidden neurons, training with ADAM)
2. ClustOfVar[10]

Similarly, we use two distinct supervised regression techniques for the reconstruction process:

[9]Self-implemented as part of the conML framework (Julia version)
[10]Implemented using https://cran.r-project.org/web/packages/ClustOfVar/index.html

| | Parameter | Setting |
|---|---|---|
| Batch Processing | UseLearnBlocks | biggest |
| | LearnBlockMinimum | 2000 |
| Construction | MinCategorySize | -1 |
| Complexity Reduction | MaxFeatures | 10 |
| | maxFilterFeatures | 50 |
| | maxFilterSamples | 15000 |
| | MaxModelReduction | false |
| Reconstruction | testSetPercentage | 0.33 |
| | MaxTestErrorAvg | 0.1 |
| | MaxTestErrorMax | 1.0 |
| | KrippendorffMetric | interval |
| | MinReliability | 0.8 |
| | AllowWeakReliability | false |

| | Parameter | Setting |
|---|---|---|
| Deconstruction | deconstructionStrategy | integrative |
| | deconstructionMode | minimal |
| | SigmaZetaCutoff | 0.2 |
| | MaxDistanceT | 1.0 |
| | FullTolerance | 0.1 |
| | TΣTolerance | 0.0 |
| | MaxTimeConflicts | 0.1 |
| Other | highestLevel | 9 |
| | usePredictionsAsTargets | false |
| | enablepile | true |
| | enablewaste | false |
| | lscvBandwidth | true |
| | learningDomain | Procedural |

**Table 2**
Settings for Extracting Procedural Knowledge with Constructivist Machine Learning. Parameters are grouped according to the step in the pipeline that is influenced. See [14] for detailed explanation.

1. Multi-Layer Perceptron[11] (number of input neuron depending on model size, one hidden layer with number of hidden neurons depending on model size, tanh activation function for hidden neurons, training with lbfgs, 3-fold cross-validation)
2. AdaBoost[12] (number of decision stumps: 10)

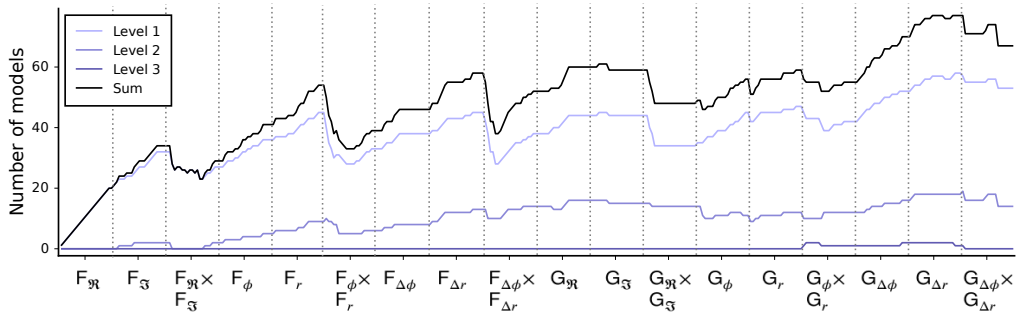
**Training Progress.** Figure 5 gives an overview of the training process. Most notably, the number of models in the knowledge base increases only slowly in the beginning but it does not saturate over the course of this training process (Figure 5a), as was observed during the conceptual experiment before (cf. Fig. 4). As a comparison of Figure 5b)-d) indicates, the growth of the procedural knowledge base speeds up after the first half of data batches with respect to the number of models on processing levels 1 and 2 as well as the total number models within the knowledge base.

Compared to the previous training process for conceptual knowledge, also a significantly larger number of models is created over the whole training process. This may in part be caused by the employing a different winner selection strategy for the procedural domain (cf. [38]), where multiple winners are now allowed in the procedural domain reconstruction if they are not overly redundant. This approach was introduced with the aim of overcoming previous limitations in the abstraction capabilities in the procedural knowledge domain (cf. [14]).

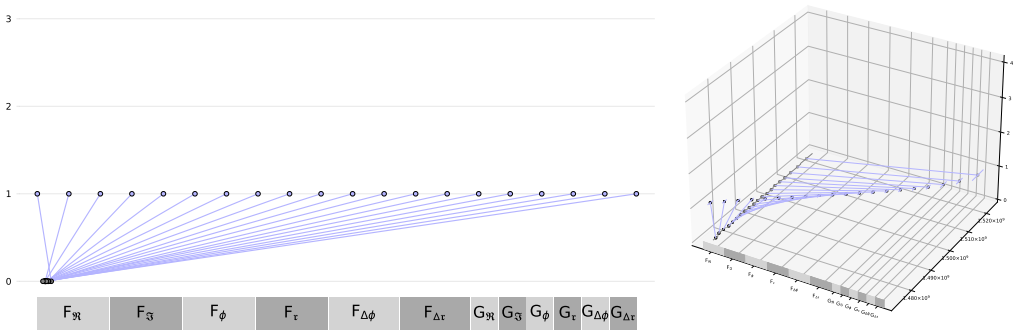
As Figures 5c) and d) illustrate, this and other algorithmic improvements – like an added rescaling step which ensures that outputs of the construction process are within reasonable a reasonable value range independently of input data or the specific algorithm – made it possible to construct procedural models not only directly on the input data, but also from the output of other machine learning models; up to processing level 3, and partially even up to level 4 (5a)). For the conceptual domain, this was already observed in previous work [14].

**Results.** After all 18 data batches have been processed, a constructivist knowledge base has been generated with 137 procedural models on processing level 1, 70 procedural models on processing level 2, and 8 procedural models on processing level 3. As during training with batches 10 through 15 also procedural models on processing level 4 could be observed (cf. 5a), this final result implies that some previously integrated models (e.g., from level 4) have been removed by means of deconstruction during the training process, as intended [18].

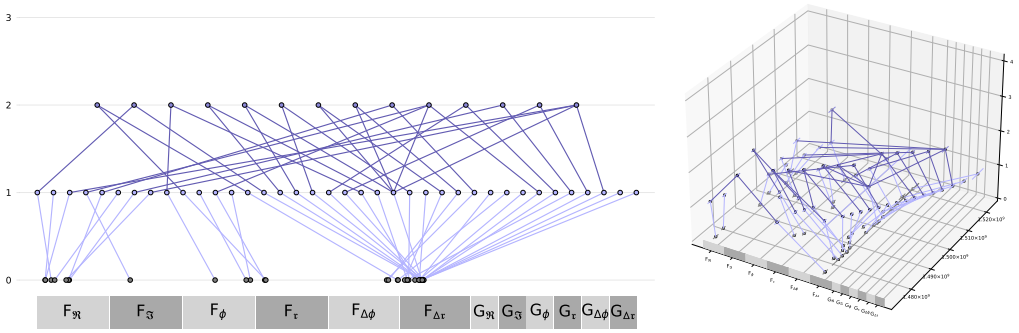
[11]Implemented using https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html
[12]Implemented using https://github.com/bensadeghi/DecisionTree.jl



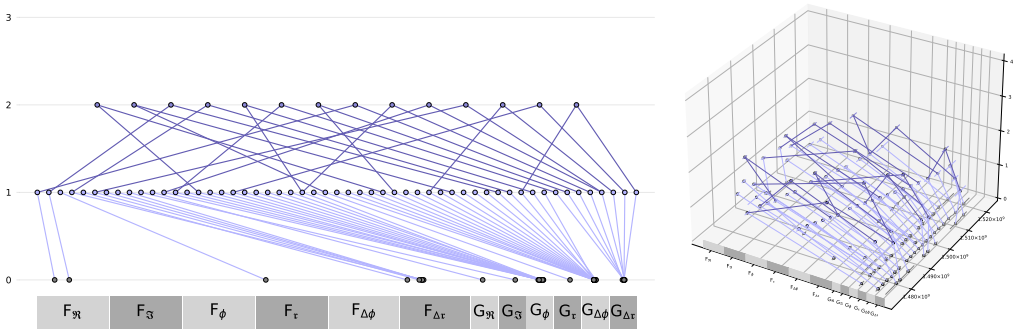

a) Knowledge base development during consecutive training over 18 data batches.



b) Knowledge base state after data batch $F_{\Re}$ (1 of 18) viewed upfront (left) and in 3D (right).

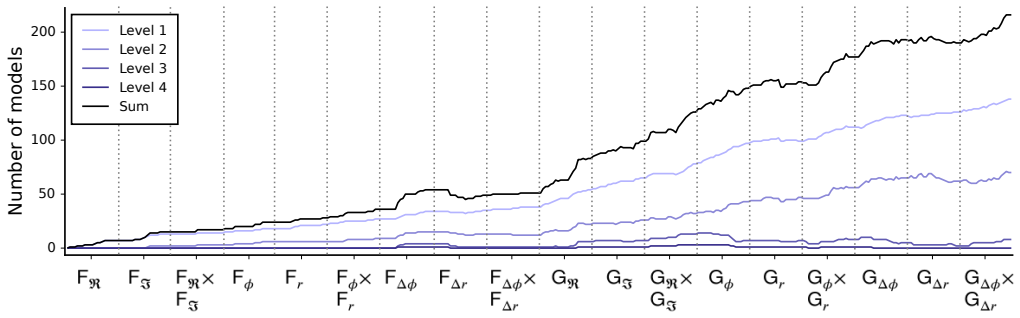


c) Knowledge base state after data batch $F_{\Delta\phi} \times F_{\Delta r}$ (9 of 18) viewed upfront (left) and in 3D (right).

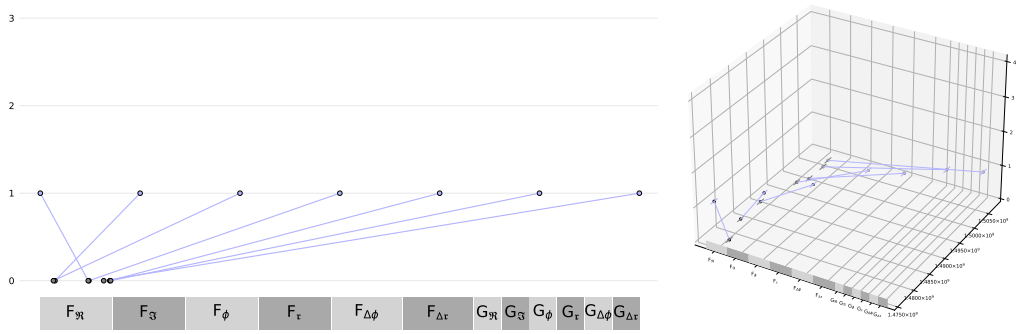


d) Knowledge base state after data batch $G_{\Delta\phi} \times G_{\Delta r}$ (18 of 18) viewed upfront (left) and in 3D (right).

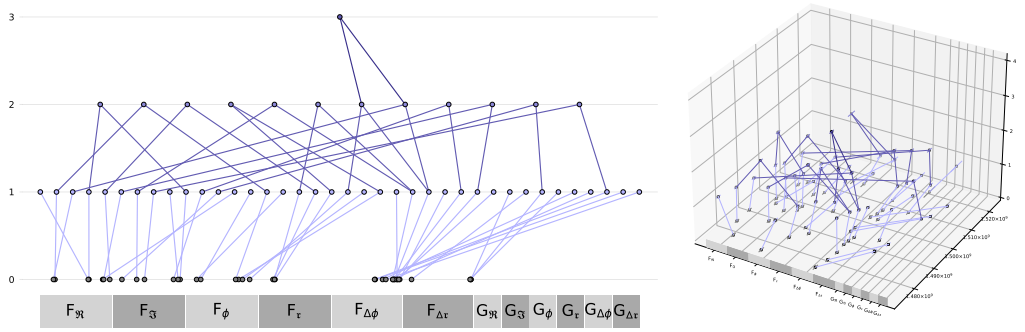
**Figure 4:** Overview of the learned knowledge during conceptual knowledge extraction. a) depicts the number of stored models over the course of the learning process. b) - d) illustrate the state of the knowledge base at selected points during the learning procedure. The 2D views on the left correspond to the 3D representations on the right (when viewed from down left). The x-axes represent the different input feature subsets in processing order from left to right and the y-axes depicts the abstraction level of each stored model. Every entry on level one or above represents a single stored model, while the connected points on the zeroth level indicate the feature subset that each model received as input. Models on higher levels abstract the knowledge from the connected models below. The z-axes always refer to the temporal limits of the stored models.



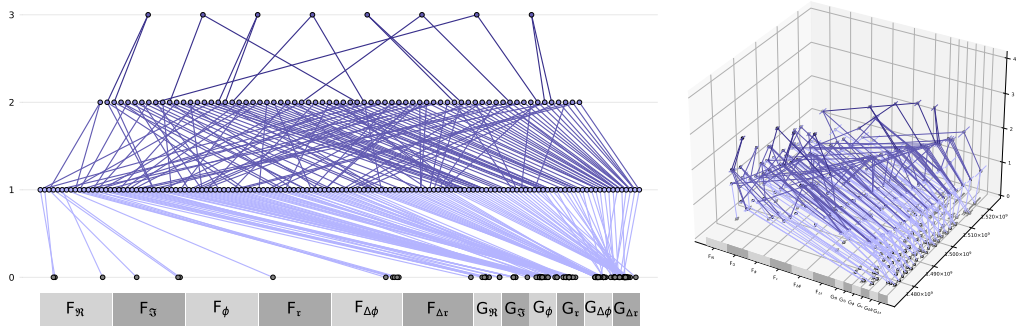

a) Knowledge base development during consecutive training over 18 data batches.



b) Knowledge base state after data batch $F_{\mathfrak{R}}$ (1 of 18) viewed upfront (left) and in 3D (right).

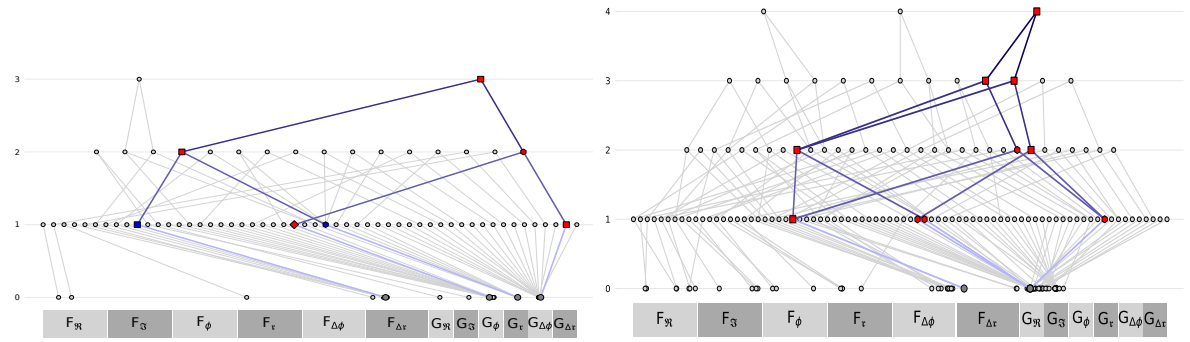


c) Knowledge base state after data batch $F_{\Delta\phi} \times F_{\Delta r}$ (9 of 18) viewed upfront (left) and in 3D (right).



d) Knowledge base state after data batch $G_{\Delta\phi} \times G_{\Delta r}$ (18 of 18) viewed upfront (left) and in 3D (right).

**Figure 5:** Overview of the learned knowledge during procedural knowledge extraction. a) depicts the number of stored models over the course of the learning process. b) - d) illustrate the state of the knowledge base at selected points during the learning procedure. The 2D views on the left correspond to the 3D representations on the right when viewed from down left. The x-axes represent the different input feature subsets in processing order from left to right and the y-axes depicts the abstraction level of each stored model. Every entry on level one or above represents a single stored model, while the connected points on the zeroth level indicate the feature subset that each model received as input. Models on higher levels abstract the knowledge from the connected models below. The z-axes always refer to the temporal limits of the stored models.



a) Conceptual knowledge base after batch 16 of 18.

b) Procedural knowledge base after batch 12 of 18.

**Figure 6:** Highlighting of selected high-level models and their dependencies. Models reconstructed by an AdaBoost algorithm are colored red, random forest models green, and models reconstructed by a MLP blue. The marker shape is associated with the purpose meta datum and reflects the kind of knowledge learned by each model. For the conceptual case in a) a square indicates that the model targets were generated by a KMeans approach whereas a hexagon indicates targets derived from Ward's hierarchical clustering. All highlighted models represent models that distinguish two classes. For the procedural models in b) a square means that the corresponding models' targets were generated by the ClustOfVar algorithm and hexagons indicate models whose targets are derived from the middle layer nodes of a sparse autoencoder network.

## 5. Discussion

With this work, we illustrate not only difference in the extraction of conceptual and procedural knowledge, but also the general outcomes of a Constructivist Machine Learning application. Therefore, we discuss not only algorithmic advances for the procedural domain, but also to what extend explainability is improved by our framework.

**Procedural Winner Selection.** Recently, we have introduced a modified winner selection mechanism for procedural knowledge [38]. Rationale behind this is that isolating a single winner candidate in the procedural domain will often discard a lot of valuable information about input data and yield a descriptive model for only a subset of its input samples. Therefore, discarding all but one candidates is not strictly necessary if a useful and coherent subset of candidates can be identified. Since the preliminary models resulting from the construction step were split to contain only one of the original target dimensions, selecting a set of winner models instead of a single one may allow us to retain more of the descriptive power of the original higher-dimensional targets. In this sense, multiple winner selection is a natural addition to the definition of learning in the procedural domain, as it is the counterpart to splitting the target dimensions in the reconstruction step.

Given that the chosen algorithms for construction are sufficiently distinct and able to capture different aspects of the input data, the final set of candidate models (prior to winner selection) is expected to predominantly consist of models that represent relationships that can be reconstructed with high intersubjectivity. A high degree of model diversity is expected, but redundant models are clearly also possible, since all model candidates are processed independently up to this point and moreover, highly redundant model candidates naturally tend to either all pass or all fail the quality control steps.

**Explainability through Temporal Validity.** The definition of a temporal validity of learned knowledge employed in our framework makes it transparent that whenever data is to be evaluated for classification or regression tasks, it must be distinguished as to whether they are covered by an already learned model or not. This can be reconciled using the meta data $T$ and can be helpful for

the interpretation of the results. For instance, while predictions for temporally covered test data are essentially interpolations of known training data, predictions based on data that are not temporally covered by learned models are extrapolations which need to be interpreted more carefully.

**Explainability through Hierarchy.** A central goal in combining machine learning and knowledge engineering is to obtain clear structures and explainable results from unstructured data. The method proposed advances towards this goal due to the fact that not only the learning processes themselves are based on human thought structures but also the representation of the learned knowledge. Learned models are distinguished by their explicitly expressed limitations of validity and inter-model connections that are derived based on principles from constructivist learning theories. In this respect, the proposed method differs from many established supervised and unsupervised learning techniques that provide users with solutions but not with a solution path that can be interpreted intuitively.

The hierarchical organization of learned models in a knowledge domain allows their actual functionality to be traced in detail. By mapping the model purpose as a pragmatic property $Z$, e.g., it is immediately determinable for each conceptual model whether it is a classifier that distinguishes between two, three, or four classes. The conceptual knowledge domain as a whole thus becomes interpretable as a connectome of classifiers that are unidirectionally linked (cf. Fig. 6a). For the procedural domain, the target generating process at each step is made transparent and the knowledge stored by high-level abstracting models is directly interpretable as a series of individual low-complexity data transformations (cf. Fig. 6b), similar to the flow of data through multi-layered neural networks. In both cases, the generated abstracting models on processing level 1 ror higher integrate diverse models with differently generated target values, different reconstruction methods and input data generated at different steps in the constructivist learning process. For the goal of explainability, this allows an intuitive backtracking of the individual solution path of a domain model, even on a case-by-case basis.

## 6. Conclusions

Building on the previously introduced concepts for constructivist machine learning [15, 17, 18], we here demonstrate how this framework may be employed in order to extract conceptual or procedural knowledge. The resulting constructivist knowledge domain explicitly maps relationships between learned models, making them immediately comprehensible. These relations are not only defined by links between inputs and outputs of the models, but especially by pragmatic properties describing temporal, subjective and purpose-related limitations. Their consistent use in the form of metadata not only improves the comprehensibility of learning and results, but also allows for an effective handling of ambiguities and operationalizing that only unambiguous models should be included in a knowledge domain. To this end, Constructivist Machine combines the strengths of machine learning with those of knowledge engineering concepts and pursues core ideas of hybrid intelligent systems. As of now, Constructivist Machine Learning does not support factual and metacognitive knowledge, which, however, we aim to integrate in future versions of the Constructivist Machine Learning framework. At the same time, we plan to investigate usage of the already available functionality to extract conceptual and procedural knowledge in further application domains.

## Download

In order to facilitate the application of Constructivist Machine Learning in practice, we implemented this concept as a multi-language framework called *conML*. Current versions for Python, R, and Julia are available as open-source software at www.constructivist.ml/download

## Acknowledgements

We would like to thank Sandra Neubert for reviewing our manuscript, Dmitrij Denisenko and Dennis Carrer for working on Python and R re-implementations of the original prototype implementation, and Michael Hermelschmidt for prototyping a framework-specific visualization tool.

## References

[1] D. Vrandecic, The rise of wikidata, IEEE Intelligent Systems 28 (2013) 90–95.
[2] D. Vrandečić, M. Krötzsch, Wikidata: a free collaborative knowledgebase, Communications of the ACM 57 (2014) 78–85.
[3] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, et al., DBpedia – a large-scale, multilingual knowledge base extracted from wikipedia, Semantic web 6 (2015) 167–195.
[4] K. Bollacker, R. Cook, P. Tufts, Freebase: A shared database of structured general human knowledge, in: 22nd AAAI Conference on Artificial Intelligence, AAAI Press, 2007, p. 1962–1963.
[5] S. Marcus, Automating knowledge acquisition for expert systems, volume 57, Springer Science & Business Media, 2013.
[6] J. H. Boose, A survey of knowledge acquisition techniques and tools, Knowledge Acquisition 1 (1989) 3–37.
[7] G. Kasneci, F. Suchanek, G. Weikum, Yago: A core of semantic knowledge (2006).
[8] F. Mahdisoltani, J. Biega, F. Suchanek, Yago3: A knowledge base from multilingual wikipedias, in: 7th biennial conference on innovative data systems research, CIDR Conference, 2014.
[9] O. Etzioni, A. Fader, J. Christensen, S. Soderland, et al., Open information extraction: The second generation, in: Twenty-Second International Joint Conference on Artificial Intelligence, 2011.
[10] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, W. Zhang, Knowledge vault: A web-scale approach to probabilistic knowledge fusion, in: Proceedings of the 20th International Conference on Knowledge Discovery and Data Mining, 2014, pp. 601–610.
[11] A. Holzinger, Introduction to machine learning & knowledge extraction (make)., Mach. Learn. Knowl. Extr. 1 (2019) 1–20.
[12] A. C. Khadir, H. Aliane, A. Guessoum, Ontology learning: Grand tour and challenges, Computer Science Review 39 (2021) 100339.
[13] A. Martin, K. Hinkelmann, A. Gerber, D. Lenat, F. van Harmelen, P. Clark, Preface, in: A. Martin, K. Hinkelmann, A. Gerber, D. Lenat, F. van Harmelen, P. Clark (Eds.), Proceedings of the AAAI 2019 Spring Symposium on Combining Machine Learning with Knowledge Engineering (AAAI-MAKE 2019), 2019. URL: http://ceur-ws.org/Vol-2350/.
[14] T. Schmid, Automatisierte Analyse von Impedanzspektren mittels konstruktivistischen maschinellen Lernens, Ph.D. thesis, Leipzig, 2018.
[15] T. Schmid, Deconstructing the final frontier of artificial intelligence: Five theses for a constructivist machine learning, in: A. Martin, K. Hinkelmann, A. Gerber, D. Lenat, F. van Harmelen, P. Clark (Eds.), Proceedings of the AAAI 2019 Spring Symposium on Combining Machine Learning with Knowledge Engineering (AAAI-MAKE 2019), 2019. URL: http://ceur-ws.org/Vol-2350/.
[16] A. Riegler, Constructivism, in: Paradigms in theory construction, Springer, 2012, pp. 235–255.
[17] T. Schmid, Using learning algorithms to create, exploit and maintain knowledge bases: Principles of constructivist machine learning, in: A. Martin, K. Hinkelmann, A. Gerber, D. Lenat, F. van Harmelen, P. Clark (Eds.), Proceedings of the AAAI 2020 Spring Symposium on Combining Machine Learning with Knowledge Engineering (AAAI-MAKE 2020), 2020.

[18] T. Schmid, Batch-like online learning for more robust hybrid artificial intelligence: Deconstruction as a machine learning process, in: A. Martin, K. Hinkelmann, H.-G. Fill, A. Gerber, D. Lenat, R. Stolle, F. van Harmelen (Eds.), Proceedings of the AAAI 2021 Spring Symposium on Combining Machine Learning with Knowledge Engineering (AAAI-MAKE 2021), 2021.
[19] A. Newell, The knowledge level, Artificial intelligence 18 (1982) 87–127.
[20] R. Davis, H. Shrobe, P. Szolovits, What is a knowledge representation?, AI Magazine 14 (1993) 17–33.
[21] L. Zagzebski, What is knowledge?, The Blackwell guide to epistemology (2017) 92–116.
[22] A. Oeberst, J. Kimmerle, U. Cress, What is knowledge? Who creates it? Who possesses it? The need for novel answers to old questions, in: Mass collaboration and education, Springer, 2016, pp. 105–124.
[23] L. R. Grimm, Psychology of knowledge representation, Wiley Interdisciplinary Reviews: Cognitive Science 5 (2014) 261–270.
[24] J. O. y Gasset, What is knowledge?, SUNY Press, 2002.
[25] M. Biggs, the concept of knowledge in art and design, Working papers in art & design 2 (2002).
[26] A. B. Markman, Knowledge representation, Psychology Press, 2013.
[27] K. D. Knorr-Cetina, The scientist as a practical reasoner: Introduction to a constructivist and contextual theory of knowledge, in: K. D. Knorr-Cetina (Ed.), The Manufacture of Knowledge, Pergamon, Amsterdam, 1981, pp. 1–32.
[28] K. Reich, Systemisch-konstruktivistische Didaktik. Eine allgemeine Zielbestimmung, Die Schule neu erfinden (1996) 70–91.
[29] L. W. Anderson, D. R. Krathwohl, A taxonomy for learning, teaching and assessing: a revision of Bloom's taxonomy of educational objectives, Longman, New York, 2001.
[30] S. Sjøberg, Constructivism and learning, International encyclopedia of education 5 (2010) 485–490.
[31] D. Rustemeyer, Konstruktivismus in der Erziehungswissenschaft, in: Stichwort: Zeitschrift für Erziehungswissenschaft, volume 2, Springer, 2013, pp. 125–144.
[32] B. S. Bloom, Taxonomy of educational objectives. Vol. 1: cognitive domain, McKay, New York, 1956.
[33] H. Sockett, Bloom's taxonomy: a philosophical critique (i), Cambridge Journal of Education 1 (1971) 16–25.
[34] R. Pring, Bloom's taxonomy: a philosophical critique (2), Cambridge Journal of Education 1 (1971) 83–91.
[35] E. J. Furst, Bloom's taxonomy of educational objectives for the cognitive domain: philosophical and educational issues, Review of Educational Research 51 (1981) 441–453.
[36] D. R. Krathwohl, A revision of Bloom's taxonomy: an overview, Theory into Practice 41 (2002) 212–218.
[37] H. Stachowiak, Allgemeine Modelltheorie, Springer, 1973.
[38] F. P. Große, A Revision of Procedural Knowledge In the conML Framework, Master's thesis, Universität Leipzig, Leipzig, Germany, 2021.
[39] K. Reich, Konstruktivistische Didaktik. Lehren und Lernen aus interaktionistischer Sicht, 2nd ed., Luchterhan, Munich, 2004.
[40] A. K. Jain, Data clustering: 50 years beyond k-means, Pattern Recognition Letters 31 (2010) 651–666.
[41] H. Peng, F. Long, C. Ding, Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy, IEEE Transactions on pattern analysis and machine intelligence 27 (2005) 1226–1238.