Batch-like Online Learning for More Robust Hybrid Artificial Intelligence:
Deconstruction as a Machine Learning Process

Thomas Schmid[a,b]

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

Abstract

Continuous streams of data are a common, yet challenging phenomenon of modern information processing. Traditional approaches to adopt machine learning techniques to this setting, like offline and online learning, have demonstrated several critical drawbacks. In order to avoid known disadvantages of both approaches, we propose to combine their complementary advantages in a novel machine learning process called deconstruction. Similar to supervised and unsupervised learning, this novel process provides a fundamental learning functionality modeled after human learning. This functionality integrates mechanisms for partitioning training data, managing learned knowledge representations and integrating newly acquired knowledge with previously learned knowledge representations. A prerequisite for this concept is that learning data can be partitioned and that resulting knowledge partitions may be accessed by formal means. In the proposed approach, this is achieved by the recently introduced Constructivist Machine Learning framework, which allows to create, exploit and maintain a knowledge base. In this work, we highlight the design concepts for the implementation of such a deconstruction process. In particular, we describe required subprocesses and how they can be combined.

Keywords

Artificial Intelligence, Online Learning, Constructivist Machine Learning

Since the early beginning of neural networks and machine learning, it has been acknowledged that data to be analyzed by learning algorithms may be subject to change over time. Classical examples are applications from the area of forecasting and time-series prediction, which have been investigated since the middle of the last century [1]. With the rise of big data and the internet of things, however, almost any data-driven company faces the challenge of how to deal with new incoming data [2]. For machine learning models for classification or regression, in particular, the question is whether data to be tested or analyzed may be at one point too different from data used for training. This phenomenon is also known as concept drift [3].

To this end, two main strategies, called offline and online learning, are applicable [4]. While offline or batch learning implies to perform training with all data given at a specific point in time, online learning implies incremental execution of a training algorithm with only a small part of data and with the option of a life-long learning process. Batch learning typically allows for easier parallelization and therefore better training performance, but requires re-training for

new data and therefore increases computational costs. Online learning allows for a straightforward handling of data streams, but may be slow and subject to unintended semantic shifts in the underlying model. New data, e.g., may induce bias into classification data and may slowly alter the model. Moreover, the question arises how to deal with data that only partially lead to good results, e.g, if training is only successful for data from a certain time span.

With such partial effects, overall performance decreases. Traditionally, learning would still be continued, and performance may even further decrease. But what if splitting the training data or omitting parts of it would increase prediction quality? In order to assess such alternative scenarios and avoid performance drawback from continuous online learning, we have recently introduced the concept of Constructivist Machine Learning [5]. Following both the philosophical paradigm of constructivism and the technological paradigm of hybrid intelligent systems, this framework allows to evaluate and automatically adjust training data partitioning [6]. As a consequence, semantic building blocks - or Stachowiak-like models [7], respectively - can be identified, related to each other in a hierarchical scheme and updated when necessary.

Here, we focus on the handling of data streams by implementing deconstruction as a machine learning process. We point out that for semantic updating in an online learning scenario, metadata is required and present an algorithmic scheme that allows to resolve disambiguity, generalize models or create abstracted models. This scheme combines supervised and unsupervised machine learning techniques and exploits temporal and other metadata. As a result, this algorithmic deconstruction process allows not only to create a hierarchical knowledge base, but in particular to handle streams of data with respect to temporal validity.

Principles of Constructivist Machine Learning

According to dominating modern educational concepts, human learning takes place through construction, reconstruction or deconstruction processes [8]. Following this paradigm, we have introduced concepts to implement such learning processes [6]. To put all three processes into practice, a corresponding knowledge base is required that consists of Stachowiak-like models and employ a data management process in order to organize for efficient learning.

Data Management. Assuming a data stream as input, the starting point for Constructivist Machine Learning is a set or batch of samples possessing pragmatic metadata (Fig. 1). Such metadata describe timely validity (T) and validity regarding subjects (Σ) and purposes (Z) [6]. From this initial batch, subsets are identified and re-grouped into learn blocks. Depending on whether samples match regarding Σ and Z, T and Σ, T and Z or all metadata ($T\Sigma Z$), ΣZ-, $T\Sigma$-, TZ- or completely related learn blocks may be identified. For efficiency, only the largest learn block of a given batch will be used. Not all forms of relationship, however, are equally suitable for a model construction. Constructions from TZ- or $T\Sigma Z$-related learn blocks, e.g., offer little added value. Learn blocks of $T\Sigma Z$-related samples that are divergent even represent a source of serious error. Learn blocks of TZ-related models allow the generation of new models, which, however, will represent an intersubjective reconstruction rather than a construction process. For constructions, learn blocks of ΣZ-related vector models are therefore preferred. Once a learn block is selected, it will undergo a sequence of learning processes. After these processes have terminated, the knowledge base is updated accordingly, which may imply storing a newly
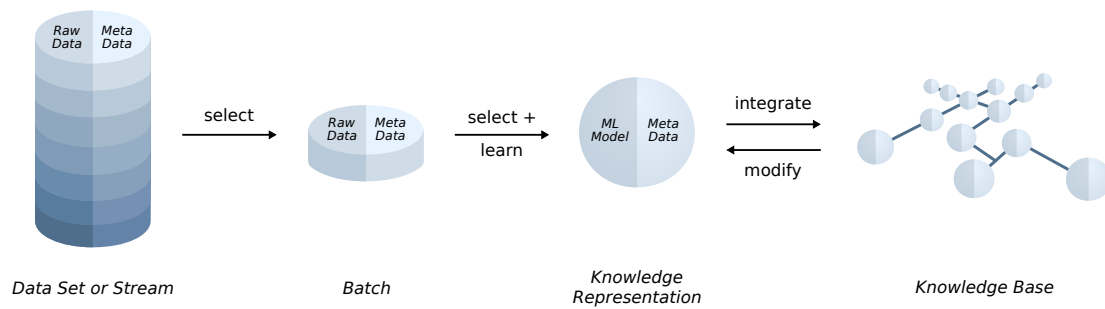
Figure 1: Transformation of real-world data into an abstract knowledge base. Using a Constructivist Machine Learning approach, real-world data is processed batch-wise by learning algorithms with the aim of identifying an optimal representation for a given block. Each representation is then integrated into an existing knowledge base consisting of previously identified representations. See also [6].

reconstructed model as well as modifying or deleting existing models from the knowledge base. If further batches exist, this sequence of selecting and processing data is repeated.

Representation Learning. Various combinations of learning processes are possible for a given learn block. If the knowledge base is still empty and target values are defined for the learn block, e.g., only a reconstruction process is carried out and the resulting model is stored in the knowledge base. In an educational context, reconstruction implies in general application, repetition or imitation, in particular the search for order, patterns or models [9, p. 145]. Similarly, the reconstruction of a machine model is here understood as supervised learning from given examples. In contrast to classical supervised learning, however, competing models are generated and evaluated for intersubjective validity. If target values are undefined for the learn block, such targets are produced in a construction process preceding the reconstruction process. In an educational context, construction is in general associated with creativity, innovation and production, and in particular with the search for new variations, combinations or transfers [9, p. 145]. For machine models, this is interpreted as an unsupervised learning that identifies or defines alternative n-dimensional outputs to a set of incomplete vector models. Thereby, competing model candidates are created that are evaluated in a following reconstruction process. Rationale behind this is that it is a priori unclear which of the models constructed from a learn block can be reconstructed with best accuracy and intersubjectivity.

Knowledge Integration. After successful reconstruction, further mechanisms are applied to the respective knowledge representation in order to manage integration into the knowledge base. In particular, a deconstruction process is carried out to avoid redundancies and contradictions, if related models exist in the knowledge base. In an educational context, deconstruction in general means the re-assessment of an already existing construct regarding incompleteness, the unforeseen and the unconscious, and in particular the search for possible omissions, simplifications, additions and criticism [9, p. 145]. In Constructivist Machine Learning, deconstruction is in particular associated with automated re-training of models and creating abstracted models, which may result in modifying or discarding models of the knowledge base.
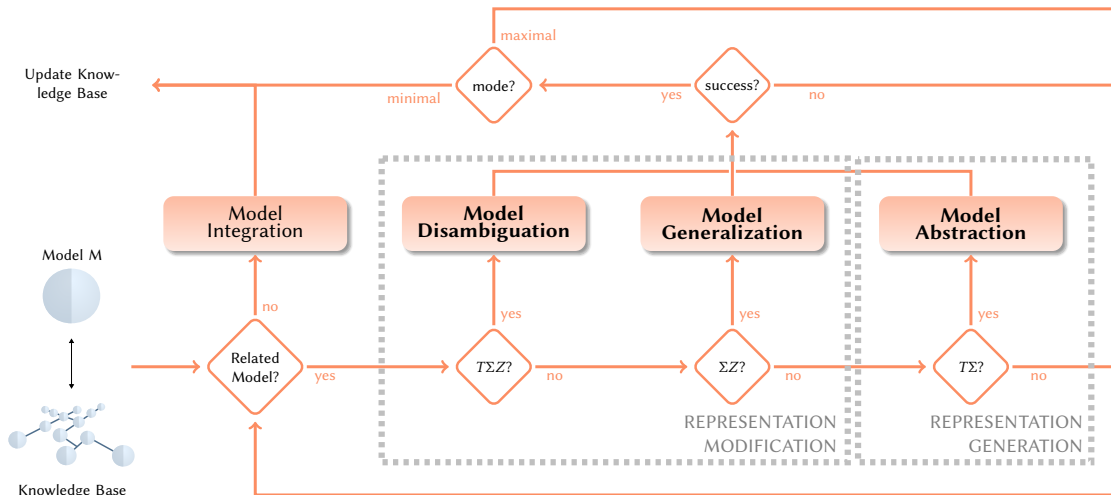
Figure 2: Overview of a machine learning-based deconstruction process. Two knowledge representations are considered. If they match, they may either undergo a representation modification or invoke the generation of a novel, abstracted knowledge representation. The subprocesses of Model Disambiguation, Model Generalization and Model Abstraction are further displayed in Figure 3, 4 and 5.

Deconstruction as a Machine Learning Process

The aim of the deconstruction process is to integrate new and old knowledge representations in a way that not only avoids ambiguity but also allows to abstract novel knowledge representations automatically. The key subprocesses of deconstruction are representation modification and representation generation (Fig. 2), of which at maximum one will be carried out for a given batch. Prerequisite for these subprocesses to be executed is that an existing knowledge representation or model, respectively, has been identified from the corresponding knowledge base that exhibits a pragmatic relationship to a newly reconstructed model. In the event that two or more related models are identified for a newly reconstructed model, they may either be deconstructed consecutively or the deconstruction process is aborted as soon as a complete, ΣZ, TZ or $T\Sigma$ deconstruction was successful.

Whether a representation modification or generation is applied to a given batch, depends on the type of relationship between the new model M and a model from the existing knowledge base. The initial task of the deconstruction process is therefore to determine this relationship. If no reationship can be identified, the new model M is integrated unaltered into the knowledge base. In case of completely and ΣZ-related models, however, this relationship is assessed by model re-training, which makes use of the reconstruction process. In case of $T\Sigma$-related models, deconstruction is carried out in terms of a knowledge abstraction procedure, which makes use of the construction process. The case of TZ-related models would reflect that models with the same purpose and same temporal validity but differing subjects have been identified, which under a fixed intersubjective reconstruction scheme is not possible; therefore, this relationship

is not explicitly handled in the following. If a newly reconstructed model shows a complete relationship to an existing model from the knowledge base, this may introduce error and contradiction into the knowledge base. Therefore, this is handled with priority.

Representation Modification. With ΣZ-related models, the aim of deconstruction is to extend or replace the existing model from the knowledge base. In particular, it is assessed whether the temporal validity of the existing model can be expanded according to the temporal validity of the new model. Both models are fused into a new model that is re-trained via the reconstruction process. If successful, the old model is replaced by the fused model, otherwise the fused model are discarded. For $T\Sigma Z$-related models, re-training may be initiated by model fusion as well as by model differentiation. Model differentiation means that it is tested whether the fused model may be split in two or more submodels of more limited temporal validity.

In contrast to ΣZ relationships, deconstruction of $T\Sigma Z$-related models can not only extend but also falsify the validity of these models. If the model fusion is falsified, the differentiation of the fused model is executed or, if necessary, one of the contradicting models is discarded. The disposal of models is carried out according to a user-defined regime, which makes a distinction between a conservative ($\mathcal{M}_{old}$ retained, $\mathcal{M}_{new}$ discarded) and an integrative ($\mathcal{M}_{old}$ discarded, $\mathcal{M}_{new}$ added to knowledge base) regime. Alternatively, if $\mathcal{M}_{new}$ is based on a larger set of vector models than $\mathcal{M}_{old}$, $\mathcal{M}_{new}$ is added to the knowledge base and $\mathcal{M}_{old}$ is discarded; otherwise, $\mathcal{M}_{old}$ is retained and $\mathcal{M}_{new}$ discarded. This regime is referred to as opportunistic.

Representation Generation. A $T\Sigma$ relationship provides the basis to construct a new model on the next higher level of the knowledge base. In this case, both models share a congruent temporal validity and a common set of model subjectives while differing in their model purpose. First, the newly reconstructed model is stored to the knowledge base. The old model from the knowledge base is left unaltered. Using the outputs, or target values respectively, of the $T\Sigma$-related models, a new learn block without target values is formed. This learn block is assigned a higher level than the underlying models possess in the knowledge base and transferred to a construction process, from which all further learning processes may be passed. Thereby, repeated abstraction from a single learn block is possible. Knowledge abstraction may be limited by a user-defined maximum of knowledge levels.

a) Model Disambiguation

A $T\Sigma Z$-based deconstruction, or model disambiguation respectively, is the most critical sub-process of deconstruction. Whenever a newly reconstructed model $\mathcal{M}$ is identified to be related to a model $\mathcal{M}_x$ within the knowledge base with respect to T as well as to Σ and Z, model disambiguation will eliminate conflicting models and assure that the knowledge base is kept free from ambiguity. In order to achieve this, $\mathcal{M}$ or $\mathcal{M}_x$ may even be disposed from the knowledge base or from being integrated to it, respectively. The three main operations of the $T\Sigma Z$-based deconstruction are model fusion, model differentation and the application of a user-defined disambiguation policy. Fig. 3 gives an overview of how they are connected with each other.

Model Fusion. With the model fusion, options are sought for merging the underlying data of the new model $\mathcal{M}$ and the $T\Sigma Z$-related $\mathcal{M}_x$ from the knowledge base. The outcome may be a fused model that is assessed by re-training, the return of the unaltered $\mathcal{M}$ to the general deconstruction process (with $\mathcal{M}_x$ left untouched) or the necessatity to resolve contradictions
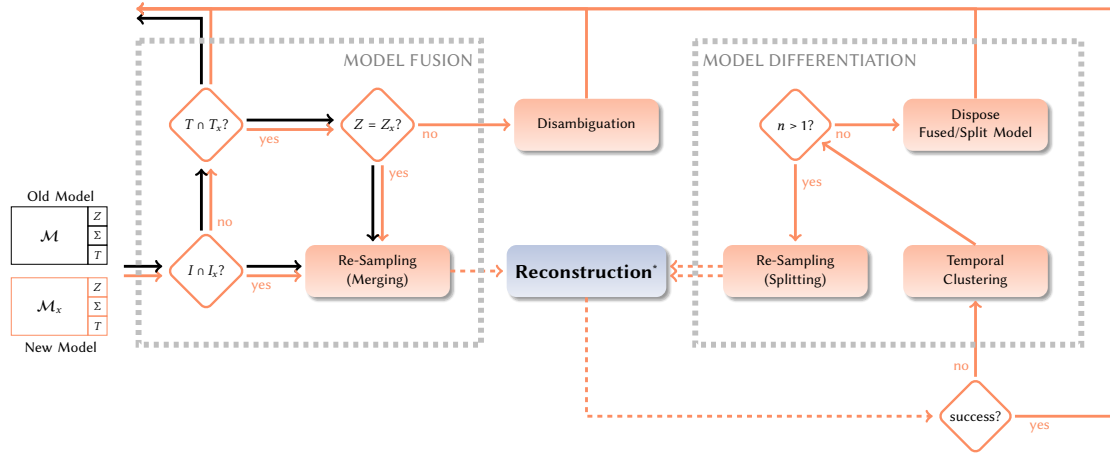
Figure 3: Schematic display of the model disambiguation or $T\Sigma Z$-based deconstruction, respectively. This subprocess is embedded within the general deconstruction process (Fig. 2). Dashed arrows indicate that this deconstruction subprocess employs a reconstruction process as defined by [6].

between $\mathcal{M}$ and $\mathcal{M}_x$. As a first step, input features are sought that are used by both $\mathcal{M}$ and $\mathcal{M}_x$. If two or more such features exist, both models are combined by reducing the underlying datasets to match the identified feature intersection and concatenating these new samples from both models to a new, fused model; this fused model will then undergo re-training through a reconstruction process as described by Schmid [6] and the originating models $\mathcal{M}$ and $\mathcal{M}_x$ as well as all models of the knowledge base depending on them will be removed. Else, a time-based intersection is sought by identifying samples from both models with matching T or an identical timestamp, respectively. If the size of such a time-based intersection does not match the minimal size requirement to form a new learn block, the new model $\mathcal{M}$ is returned untouched to the general deconstruction process (Fig. 2). If enough samples with matching timestamp are identified, re-sampling may be applied to form an alternative model. Prerequisite for this, however, is that Z of $\mathcal{M}$ and Z of $\mathcal{M}_x$ – for the given timestamps of the matched samples – are in good agreement, which is quantified by determining Krippendorff's α. In case the agreement is not sufficient, the ambiguity inherent to these two contradicting models is resolved by a user-defined disambiguation strategy. Otherwise, re-sampling is applied and model re-training will be carried out by employing a reconstruction process. The originating models $\mathcal{M}$ and $\mathcal{M}_x$ as well as all models depending on them will be removed from the knowledge base before the fused model enters the reconstrucion process.

Disambiguation. By disambiguation, it is decided which of the contradicting models $\mathcal{M}$ and $\mathcal{M}_x$ will be part of the knowledge base. Depending on user preferences, one of three disambiguation strategies are applied: conservative, integrative or opportunistic. A conservative strategy will keep $\mathcal{M}_x$, which is already part of the knowledge base, and will dispose the new $\mathcal{M}$. An integrative strategy will proceed the other way round and dispose $\mathcal{M}_x$ (and all models depending on it hierarchically) from the knowledge base while integrating the new $\mathcal{M}$ into

the knowledge base before returning to the general deconstruction process. Using an opportunistic strategy, the less intersubjective model, i.e. the model yielding a lower value for the inter-rater reliability coefficient Krippendorff's α, will be disposed while the larger model will be part of the knowledge base when returning to the general deconstruction process (Fig. 2).

Model Differentiation. If model re-training by reconstruction fails, the fused model (as well as $\mathcal{M}$ and $\mathcal{M}_x$) is regarded falsified. It will, however, not be disposed immediately. Instead, it is assessed whether it is possible to cluster the underlaying data into two or more temporally defined sublearnblocks. Following the previously defined ideas of learnblock identification [6], the number of clusters for T is determined via density estimation. Actual clusters are then identified via one-dimensional clustering, which is known to allow for optimal convergence [10]. If two or more learnblocks are identified (n > 1), re-sampling is applied to form new model candidates that will undergo re-training by reconstruction. Where only one temporal cluster is identified, the fused model (or the previously split model, respectively) is disposed; in particular, it is not returned to the general deconstruction process (Fig. 2).

b) Model Generalization

A ΣZ-based deconstruction, or model generalization respectively, is a deconstruction subprocess that aims at extending existing models regarding their temporal validity. If a newly reconstructed model $\mathcal{M}$ is identified to be related to a model $\mathcal{M}_x$ within the knowledge base with respect to Σ and Z, the model generalization will try to combine both models. In particular, it is assessed whether they can be fused into a single model with a timespan T covering the timespans of $\mathcal{M}$ and $\mathcal{M}_x$. The main operations of the ΣZ-based deconstruction are model fusion, disposal of the fused model and replacement of the old model with the fused model. Fig. 4 gives an overview of how these operations are connected with each other.

Model Fusion. During model fusion, the possibility of merging the underlying data of the old model $\mathcal{M}_x$ and the new model $\mathcal{M}$ along the temporal dimension T is investigated. The outcomes of this operation may be either a novel, larger model representing more samples (or a larger timespan, respectively) or the return to the general deconstruction process (Fig. 2) with $\mathcal{M}$ being integrated into the knowledge base and $\mathcal{M}_x$ left unaltered. Prerequisite for merging $\mathcal{M}$ and $\mathcal{M}_x$ is that an intersection of the respective input spaces I and I_x is possible. Moreover, this intersection should contain more than one feature in order to avoid trivial models. If no appropriate feature intersection can be identified, $\mathcal{M}$ and $\mathcal{M}_x$ will be part of the knowledge base. If the model fusion can be completed, on the other hand, the fused model will be assessed by employing a reconstruction process [6]. At the end of this model fusion, $\mathcal{M}_x$ is still part of the knowledge base, while $\mathcal{M}$ and the fused model are still kept in competition to each other, until the reconstruction process is completed and the winner model is selected from $\mathcal{M}$ and the fused model. Depending on the success of the reconstruction process, either $\mathcal{M}$ or the fused model will be integrated into the knowledge base.

Disposal of the Fused Model. If the reconstruction of the fused model cannot be completed succesfully, it will be disposed. In particular, it will not be integrated into the knowledge base. Instead, $\mathcal{M}$ will be integrated into the knowledge base. The old, ΣZ-related model is left unaltered and as part of the knowledge base.

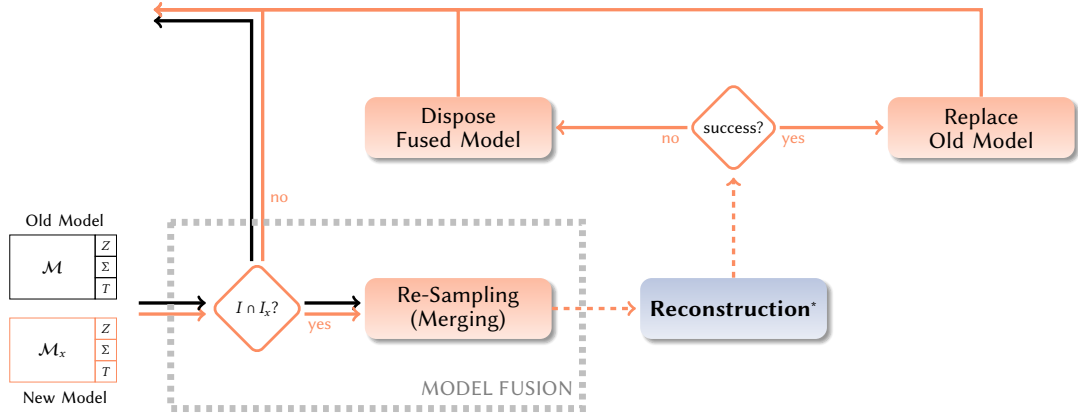**Replacement of the Old Model.** If the reconstruction of the fused model is successful, it

Figure 4: Schematic display of the model generalization or ΣZ-based deconstruction, respectively. This subprocess is embedded within the general deconstruction process (Fig. 2). Dashed arrows indicate that this deconstruction subprocess employs a reconstruction process as defined by [6].

will be integrated into the knowledge base while $\mathcal{M}$ and $\mathcal{M}_x$ will be disposed. In particular, not only $\mathcal{M}_x$, but also all models on higher hierarchy levels that depend on $\mathcal{M}_x$ will be disposed.

c) Model Abstraction

A $T\Sigma$-based deconstruction, or model abstraction respectively, is a subprocess of the deconstruction process aiming at creating novel models of a higher hierarchy order or higher abstraction level, respectively. If a newly reconstructed model $\mathcal{M}$ is identified to be related to a model $\mathcal{M}_x$ within the knowledge base with respect to T and Σ, the model abstraction will try to create novel models on the next higher level of abstraction. In particular, this new model or models are fed as model candidates into the general process of Constructivist Machine Learning [6], which constitutes from interconnected processes of construction, reconstruction and deconstruction. The main operation of the $T\Sigma$-based deconstruction is the creation of a novel learnblock that subsequently undergoes a construction process as described by [6]. Fig. 5 gives an overview of this subprocess and its operations.

Learnblock Generation. Combining two $T\Sigma$-related models differs significantly from the model fusion operations for $T\Sigma Z$- and ΣZ-related models. First of all, both the recently reconstructed model $\mathcal{M}$ and the existing $\mathcal{M}_x$ from the knowledge base are under no circumstances altered. $\mathcal{M}_x$ is left within the knowledge base, $\mathcal{M}$ is integrated unaltered into it. Instead of fusing models, a novel learnblock is generated - if the hierarchy level L of $\mathcal{M}$ matches the hierarchy level L_x of $\mathcal{M}_x$. As a user-defined minimum number of matching samples is required to form a novel learnblock, the number of underlying data for $\mathcal{M}$ and $\mathcal{M}_x$ with identical timestamps is determined. If this number of matching samples is sufficient, the similarity of the corresponding outputs of $\mathcal{M}$ and $\mathcal{M}_x$ for the identified timestamps is assessed by Krippendorff's α. If they yield no significant match, these output features are combined with the timestamps into a novel learnblock with unknown Σ and unknown Z. In particular, Σ and Z are filled with place-
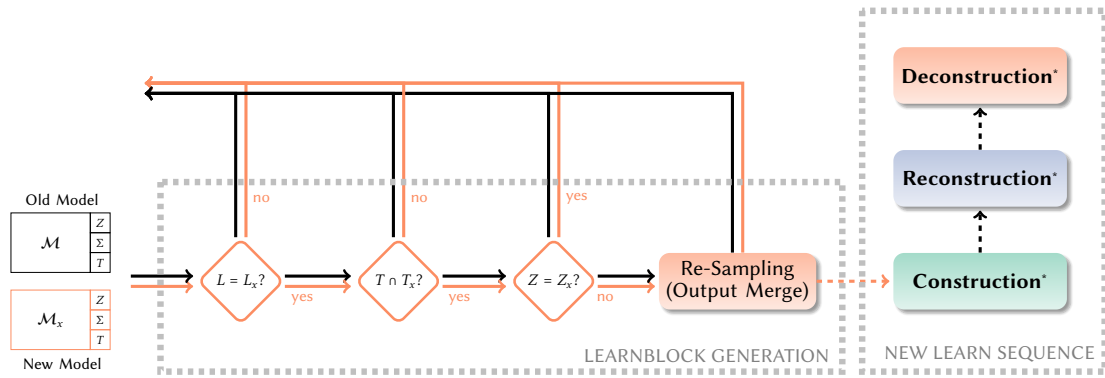
Figure 5: Schematic display of the model abstraction or $T\Sigma$-based deconstruction, respectively. This subprocess is embedded within the general deconstruction process (Fig. 2). Dashed arrows indicate that this deconstruction subprocess invokes a cycle of construction, reconstruction and deconstruction processes (as defined by [6]) for a newly generated learnblock.

holders and treated as regular learnblock in the following. The level of hierarchy, however, will of course be not n, but $n+1$ where $\mathcal{M}$ and $\mathcal{M}_x$ are located on level n.

New Learn Sequence. The $T\Sigma$-based deconstruction is basically completed successfully when a new learn block has been generated. Yet, this new learn block will be further processed in a newly entered general deconstruction process (Fig. 2). In particular, the newly generated learn block undergoes the sequence of construction, reconstruction and deconstruction processes described by Schmid [6]. Processing the new learn block in a new cycle of learning processes may be implemented as an independent learnblock cycle or in a recursive manner.

Discussion

We review advances and challenges of the concept of deconstruction as an alternative to traditional online learning. In particular, we discuss handling concept drifts and out-of-distribution effects. Finally, we sketch the relationship of this approach to knowledge engineering.

Concept Drift and Out-of-Distribution Detection. Data for given machine learning task changing over time may render a model built on old data inconsistent regarding new data. This phenomenon is known as concept drift and has been known among for decades [11]. Even more challenging than sudden concept drifts are so-called gradual drifts that evolve slowly [12]. To this end, the batch-like online learning implemented by Constructivist Machine Learning provides a more robust approach. In particular, consistency is sought not only within all individual batches being processed, but also in terms of temporal generalization. If a gradual concept drift does not allow for a long-term generalization, the approach will keep short-term or even batch-wise knowledge representations only. This approach follows the ideas of classical concepts of instance selection in order to avoid concept drifts [13, 11, 14]. More recently, the goal to avoid concept drift in practice has been approached by so-called out-of-distribution detection. The

underlying idea is to assess test data before the actual testing and reject testing if the data does not match this data distribution of the model. This approach has be widely investigated for neural networks [15, 16, 17], where many limitations for this have been found, too [18]. Due to the fact that Constructivist Machine Learning employs metadata to determine feasibility of testing data, out-of-distribution detection may be omitted.

Automated Knowledge Base Management. In the context of knowledge engineering, a knowledge representation is typically a mathematical formalization like a logic, rule, frame or semantic net related to real-world aspects [19]. Organizing such representations in a knowledge base is a central, yet time-consuming task in knowledge engineering. Managing knowledge bases may be described by typical life cycle phases. Following Martinez-Gil (2015), a creation phase is characterized by acquisition, representation, storage and manipulation of knowledge, while an exploitation phase focuses on knowledge reasoning, retrieval and sharing; the maintenance phase is concerned with integration, validation and meta-modeling of knowledge. Most work on operating knowledge bases use a semi-automated approach, leaving much space for more effective and efficient automated management strategies [20]. To this end, Constructivist Machine Learning provides a scalable approach for automatic generation of knowledge bases. Moreover, with the implementation of an algorithmic deconstruction process, it also provides automatic selection, combination and/or tuning of maintenance strategies.

Conclusions

After introducing our constructivist approach to the machine learning and knowledge engineering community [5] and outlining how to operationalize this idea [6], we have now further specified the implementation of the deconstruction process. This process is central to Constructivist Machine Learning as it is lays the foundation of interpretable and modular machine learning. Deconstruction employs existing machine learning techniques in order to integrate knowledige representations into or modify knowledge representations from a knowledge base. By combining machine learning and knowledge engineering concepts, Constructivist Machine Learning pursues ideas of hybrid intelligent systems. Yet, the ability to operate a knowledge base automatically is a significant step ahead. To this end, the deconstruction process is a key concept, which will be further investigated and assessed on real-world datasets.

Download

In order to facilitate the application of Constructivist Machine Learning in practice, we implemented this concept as a multi-language framework called *conML*. The current version is available as open-source software at www.constructivist.ml/download

Acknowledgements

References

[1] S. Makridakis, A survey of time series, International Statistical Review 44 (1976) 29–70.

[2] A. L'Heureux, K. Grolinger, H. F. Elyamany, M. A. M. Capretz, Machine learning with big data: Challenges and approaches, IEEE Access 5 (2017) 7776–7797.

[3] C. Parker, Unexpected challenges in large scale machine learning, in: Proceedings of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications, 2012, pp. 1–6.

[4] B. Pérez-Sánchez, O. Fontenla-Romero, B. Guijarro-Berdiñas, A review of adaptive online learning for artificial neural networks, Artificial Intelligence Review 49 (2018) 281–299.

[5] 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/.

[6] 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.

[7] H. Stachowiak, Allgemeine Modelltheorie, Springer, 1973.

[8] K. Reich, Systemisch-konstruktivistische didaktik. eine allgemeine zielbestimmung, Die Schule neu erfinden (1996) 70–91.

[9] K. Reich, Konstruktivistische Didaktik. Lehren und Lernen aus interaktionistischer Sicht, 2nd ed., Luchterhan, Munich, 2004.

[10] H. Wang, M. Song, Ckmeans. 1d. dp: optimal k-means clustering in one dimension by dynamic programming, The R journal 3 (2011) 29.

[11] G. Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, Machine Learning 23 (1996) 69–101.

[12] K. Stanley, Learning concept drift with a committee of decision tree, Technical Report UT-AI-TR-03-302, Department of Computer Sciences, University of Texas at Austin, Austin, Texas, USA, 2003.

[13] M. Kubat, G. Widmer, Adapting to drift in continuous domains, Technical Report ÖFAI-TR-94-27, Austrian Research Institute for Artificial Intelligence, Vienna, Austria, 1994.

[14] M. Salganicoff, Tolerating concept and sampling shift in lazy learning using prediction error context switching, in: Lazy learning, Springer, 1997, pp. 133–155.

[15] Y. Bengio, F. Bastien, A. Bergeron, N. Boulanger-Lewandowski, T. Breuel, Y. Chherawala, M. Cisse, M. Côté, D. Erhan, J. Eustache, et al., Deep learners benefit more from out-of-distribution examples, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, 2011, pp. 164–172.

[16] T. DeVries, G. W. Taylor, Learning confidence for out-of-distribution detection in neural networks, arXiv preprint arXiv:1802.04865 (2018).

[17] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. Depristo, J. Dillon, B. Lakshminarayanan, Likelihood ratios for out-of-distribution detection, in: Advances in Neural Information

Processing Systems, 2019, pp. 14707–14718.

[18] P. Kirichenko, P. Izmailov, A. G. Wilson, Why normalizing flows fail to detect out-of-distribution data, Advances in neural information processing systems 33 (2020).

[19] R. Davis, H. Shrobe, P. Szolovits, What is a knowledge representation?, AI Magazine 14 (1993) 17–33.

[20] J. Martinez-Gil, Automated knowledge base management: A survey, Computer Science Review 18 (2015) 1–9.