A Comprehensive Workplace Environment based on a Deep Learning Architecture for Cognitive Systems

Using a multi-tier layout comprising various cognitive channels, reflexes and reasoning

Thorsten Gressling

ARS Computer und Consulting GmbH
Munich, Germany
e-mail: thorsten.gressling@ars.de

Veronika Thurner

Munich University of Applied Sciences
Department of Computer Science and Mathematics
Munich, Germany
e-mail: veronika.thurner@hm.edu

***Abstract*—Many technical work places, such as laboratories or test beds, are the setting for well-defined processes requiring both high precision and extensive documentation, to ensure accuracy and support accountability that often is required by law, science, or both. In this type of scenario, it is desirable to delegate certain routine tasks, such as documentation or preparatory next steps, to some sort of automated assistant, in order to increase precision and reduce the required amount of manual labor in one fell swoop. At the same time, this automated assistant should be able to interact adequately with the human worker, to ensure that the human worker receives exactly the kind of support that is required in a certain context. To achieve this, we introduce a multilayer architecture for cognitive systems that structures the system's computation and reasoning across well-defined levels of abstraction, from mass signal processing up to organization-wide, intention-driven reasoning. By partitioning the architecture into well-defined, distinct layers, we reduce complexity and thus facilitate both the implementation and the training of the cognitive system. Each layer comprises a building block that adheres to a specific structural pattern, consisting of storage, processing units and components that are used for training. We incorporate ensemble methods to allow for a modular expansion of a specific layer, thus making it possible to introduce pre-trained functional blocks into the system. In addition, we provide strategies for generating synthetical data that support the training of the neural network parts within the processing layers. On this basis, we outline the functional modules of a cognitive system supporting the execution of partially manual processes in technical work places. Finally, a prototypical implementation serves as a proof of concept for this multilayer architecture.**

Keywords–Cognitive system; Multilayer architecture; Technical work place; Machine learning; Ensemble averaging; Synthesized training data; Context sensitive; Neural network.

I. Motivation

Many technical work places, such as laboratories or test beds, are the setting for series of well-defined, repetitive actions requiring high precision in their execution, as well as extensive documentation of every process step. Both are necessary to ensure accuracy on the one hand, and on the other hand to support accountability that often is required by law, science, or both. Typical examples are laboratories for micro-biological analysis or chemical experiments, premises of optometrists or hearing aid acousticians, or test beds for the quality inspection of produced goods, such as measuring vehicle exhaust fumes or assessing nutritional values of food.

All these settings share a number of commonalities. For one thing, within each of these working settings a human being interacts extensively with technical devices, such as measuring instruments or sensors. For another thing, processing follows a well-defined routine, or even precisely specified interaction protocols. Finally, to ensure that results are reproducible, the different steps and achieved results usually have to be documented extensively and in a precise way.

Especially in scenarios that execute a well-defined series of actions, it is desirable to delegate certain routine tasks, such as documentation or preparatory next steps, to some sort of automated assistant, in order to increase precision and reduce the required amount of manual labor in one fell swoop. At the same time, this automated assistant should be able to interact adequately with the human worker, to ensure that the human worker receives exactly the kind of support that is required in a certain context. To achieve this, the assistant needs to be context aware, i.e., equipped with cognitive input channels. As well, the assistant is expected to learn new behavioral patterns from previous experience. Thus, training the assistant appropriately is highly relevant for ensuring that the assistant's contribution really is beneficial to the human worker that it supports.

Traditional software systems for process control or workflow management are well able to support a set of well-defined processes that has been explicitly specified in advance. However, in situations that were not foreseen initially, or that were not explicitly specified because they were deemed to be highly unlikely to happen, these systems quickly meet their limits, requiring human take-over and problem solving abilities in expert mode.

The increasing capabilities of cognitive systems imply the potential for a new generation of systems that offer context sensitive reasoning on a scale hitherto unknown in machines and software systems. We exploit these possibilities by devising a cognitive hardware-software-system for supporting the execution of hybrid (i.e., partly manual and partly automated) processes in technical environments, in a manner that combines the respective strengths of human-expert-like cognition and reasoning with machine-like processing power, to improve performance, efficiency, accuracy and security issues.

By *cognitive system*, we denote a system that is capable of sensory perception and of expressing itself via technical devices, analogously to corresponding abilities found in biological organisms. Furthermore, it comprises an internal representation that is comparable to emotions. However, as system boundaries and internal states of an artificial intelligence differ greatly from those of humans beings and animals, its underlying system of values and beliefs in general differs from that of biological organisms.

After this initial motivation, we review related literature and briefly sketch the goals of our research in Section II. In Section III, we introduce the physical architecture of our cognitive system, followed by a logical architecture in Section IV. The core element of the logical architeture are building blocks, whose structure is presented in Section V. Section VI discusses approaches for effectively training the system. To illustrate the processing of our cognitive system and the interaction of the building blocks on the different layers, we present an example execution in Section VII. Section VIII sketches the prototypical implementation that we realized as a proof of concept. Finally, we critically discuss our work in Section IX, before summarizing it in Section X.

II. State of the Art and Goals

Research has already elicited several aspects that are relevant in this context. In [1], an initial version of a multilayer architecture for cognitive systems is introduced, which is enhanced here by insights into the training of the different layers that the architecture comprises. As well, the usage of ensembles is considered here, in order to improve training performance and prepare for neural reasoning.

An overview of existing approaches to computation and information architecture is provided by [2], distinguishing among others the different types of information that are processed, from subsymbolic computation focusing on data and signal processing to symbolic computation that processes data structures, thus reflecting different levels of abstraction. In [3], patterns for cognitive systems are investigated into, focusing on systems that process textual information, yet indicating that other kind of information, such as cognitively interpreted sensory data, will be addressed by cognitive systems in the near future, thus entering into new dimensions of machine cognition.

Approaches for systematic process support through Context Aware Assistive Systems (CAAS) are discussed in [4] [5] [6], in the context of manufacturing on the shop floor and human interaction with the production line. Identifying human actions observed via cameras and relating them to the manufacturing process are a crucial issue in these Context Aware Assistive Systems.

The research from [7] [8] [9] focusses on the usage of augmented reality in intelligent assistant systems, discussing among others digital projections into the current working situation, to guide the human workers through their share of the working process.

Anderson et al. [10] introduce the cognitive architecture ACT-R, which implements artificial intelligence in a symbolic way. In contrast to this, in our approach we combine symbolic and connectionistic aspects.

So far, existing supportive systems for processes that are partially executed manually in technical work places are realized mainly in a rule oriented way and implemented by algorithms. As a consequence, they cover only those situations, states and actions that have been anticipated in advance. However, they only have limited ability to learn from experience.

Recently, research on context awareness significantly progressed towards identifying a specific situation from a predefined set of possibilities in a given context and well-defined surroundings, incorporating cognition and artificial reasoning on a single level of abstraction. This single level of abstraction is then realized as a monolithic block of neural networks. However, this monolithic block needs to deal with the entire complexity by itself, which would require an extreme amount of training that exceeds what can be handled even by modern hardware.

Therefore, as a next step, we introduce an architecture for cognitive systems that expands cognition and reasoning across several levels of abstraction, to support a wide range of assistant services ranging from small actions to strategic processes. By partitioning the systems's cognition and reasoning into separate levels of abstraction (rather than implementing them as a single monolithic block), we reduce both implementation complexity and training effort. Thus, it is possible to tackle even very complex problems, which would exceed the capacity of a monolithic approach. As well, each building block of the architecture comprises components that are designated for training purposes, i.e., for generating synthetical training data and using this data to calibrate the neural network parts of the processing component.

We discuss the applicability of our approach in the context of a cognitive system that supports hybrid processes involving manual tasks within technical surroundings.

III. Physical Architecture

Physically, a technical work place comprises a variety of technical devices, as a relevant tool set to execute, or support the execution of, actions involved in the processes at the work place. Typical examples for, e.g., a chemical laboratory are electronic high precision scales, a centrifuge, a power supply or a fume hood. Some of these devices are connected to computational hardware (e.g., a remote server), either directly or via a data network. In contrast to this, other devices such as a traditional heater, operate in an isolated way, without any direct data exchange with the computational hardware. Furthermore, the work place comprises a variety of tools and devices for manual tasks, such as pipettes or glassware, as well as other materials, e.g., chemical or biological substances to be processed or analyzed.

In addition, to evolve from a technology interspersed work place towards an intelligent assistant, the work place must be equipped with devices that enable the system's cognition, as well as its interaction with the human user that executes the manual process steps. Traditionally, cognitively exploitable input devices are, for example, a microphone (with subsequent speech analysis) or a camera (with subsequent image processing). In addition to this traditional notion of audio-visual cognition, sensors and other technical measuring devices provide additional cognitive channels that supply the system with information on the current situation at the work place.

Communication from the system towards the human user is realized, e.g., via a monitor, a loudspeaker or a projector that focusses its beam of light on the tool to be used next, or that displays instructions on the process step that should be executed next. Other, more sophisticated devices arise continuously, such as mixed reality smart glasses for displaying instructions directly into the field of vision, activity tracking bracelets that combine skin and body sensors with functionality for alerting its wearer, or even EEGs for integrating information on the human user's brain activity into the system's data pool.

Note that some of these technical devices may include their own data storage, as well as computational hardware, thus being able to directly aggregate and process the data they collected, before passing it on to more sophisticated computational hardware for integration with the data from other devices and subsequent further processing.

IV. Logical Architecture

The cognitive system that we devise to support process execution is embedded into this technical work place.

Logically, we design a multilayer architecture that structures the cognitive system into different levels of abstraction (see Figure 1), rising from concrete at the bottom towards more and more abstract as we move upwards on the processing level stack. Thus, each layer Mj encapsulates processing on a specific level of abstraction, and focuses on different tasks. By structuring the overall system into logical processing layers, it is possible to train each layer individually for its respective tasks. Furthermore, modularizing the overall system improves performance by reducing processing time, as the different layers can be run in parallel.

Processing involves the analysis of incoming data, which is synthesized and analyzed to identify the situation that the work place is in, corresponding to an overall system state in the context of the executed processes. From the identified situation, processing derives, which actions should be taken as next steps, and passes these on as instructions to other layers, systems, technical devices or – via output devices – to the human user. Thus, the cognitive system is able to effectively support the human user in a context sensitive way. Note that these suggested actions are determined by aggregated conclusions that the system draws from its analysis.

Layers are interconnected by communication channels. Note that the information on situations flows upwards in the processing layer stack via channels S (white block arrows in Figure 1), whereas instructions are passed down from layer to layer via channels I (black block arrows).

The processing layer stack is based on a layer of technical devices $D1, \ldots, Dm$ as described above. These devices collect data on the work place and enter these into the cognitive system as situation information via channels $S1.i$, with $1 <= i <= m$. Some of these devices (such as Dm in Figure 1) merely collect data, e.g., by simple measuring, and pass them straight on to the first processing layer $M1$. Other devices (such as $D1$ and $D2$ in Figure 1) comprise an independent processing component ($M0.1$ or $M0.2$, respectively), which preprocesses the data before entering it into the first processing layer.

Moving upwards on the processing layer stack, situation information is aggregated from separate small snippets of measured data into larger contexts, such as actions, sequences of actions or even entire processes. Analogously, abstract instructions that are passed from top to bottom are made more and more specific from layer to layer, down to signals that operate a specific technical device in the bottom layer.

On each layer, processing takes into account the situation information that is entered into the layer from below, as well as the instruction information that is passed to the layer from above. Thus, situations are interpreted in the light of instructions that reflect the larger context of the overall system, as identified on the higher levels of abstraction.

As depicted for layer Mj in Figure 1, a processing layer can merge several process layer stacks, each representing a different work place. Thus, their information flows are integrated and consolidated, allowing for integrated information processing on a cross-organizational level of abstraction.

V. Building Blocks

Each layer in Figure 1 is implemented by a building block that follows the architectural pattern depicted in Figure 2 for a building block i, with $0 <= i <= n$, in a cognitive system that comprises $n >= 1$ processing layers stacked on top of one layer of technical devices.

Building block i is linked with the building blocks of its surrounding processing levels $i-1$ and $i+1$ via communication channels, depicted as block arrows in Figure 2. Thus, the situation information perceived by building block $i - 1$ is passed on via channel Si to building block i, which stores the information in its storage for situations. Analogously, instructions issued by building block $i + 1$ are passed on via channel $I(i + 1)$ to building block i, which stores the information in its storage for instructions.

The central part of each building block is its processing unit, which comprises both aspects of algorithmic logic and of artificial intelligence (implemented via one or more neural networks), in varying proportions (see Figure 3). On the lower levels of abstraction, the major part of processing is accomplished by algorithmic logic, whereas on the higher levels of abstraction, aspects of artificial intelligence dominate the processing.

The processing unit works on three different kinds of input:

- $E1$: Information on situations
- $E2$: Information on instructions
- $E3$: Relevant general factual knowledge, stored as rules in the knowledge database

Based on these inputs, the processing unit analyses the incoming information, interprets it and synthesizes it into its interpretation of the situation, thus lifting the previous information on the situation onto a higher level of abstraction. For example, on layer $M1$, short snippets of data that were measured by the technical devices (e.g., an electronic scale and a heater) in the underlying physical layer are gathered, consolidated and then passed on to layer $M2$. On layer $M2$, this consolidated data is then merged and interpreted to build a larger semantic context, e.g., a certain step in a chemical experiment where a certain amount of substance must be added to an existing mixture, and then heated to a specific temperature. In order to properly identify the semantic context correctly, the processing unit in layer $M2$ incorporates known "recipes" of chemical experiments that are stored within the knowledge database of layer $M2$.

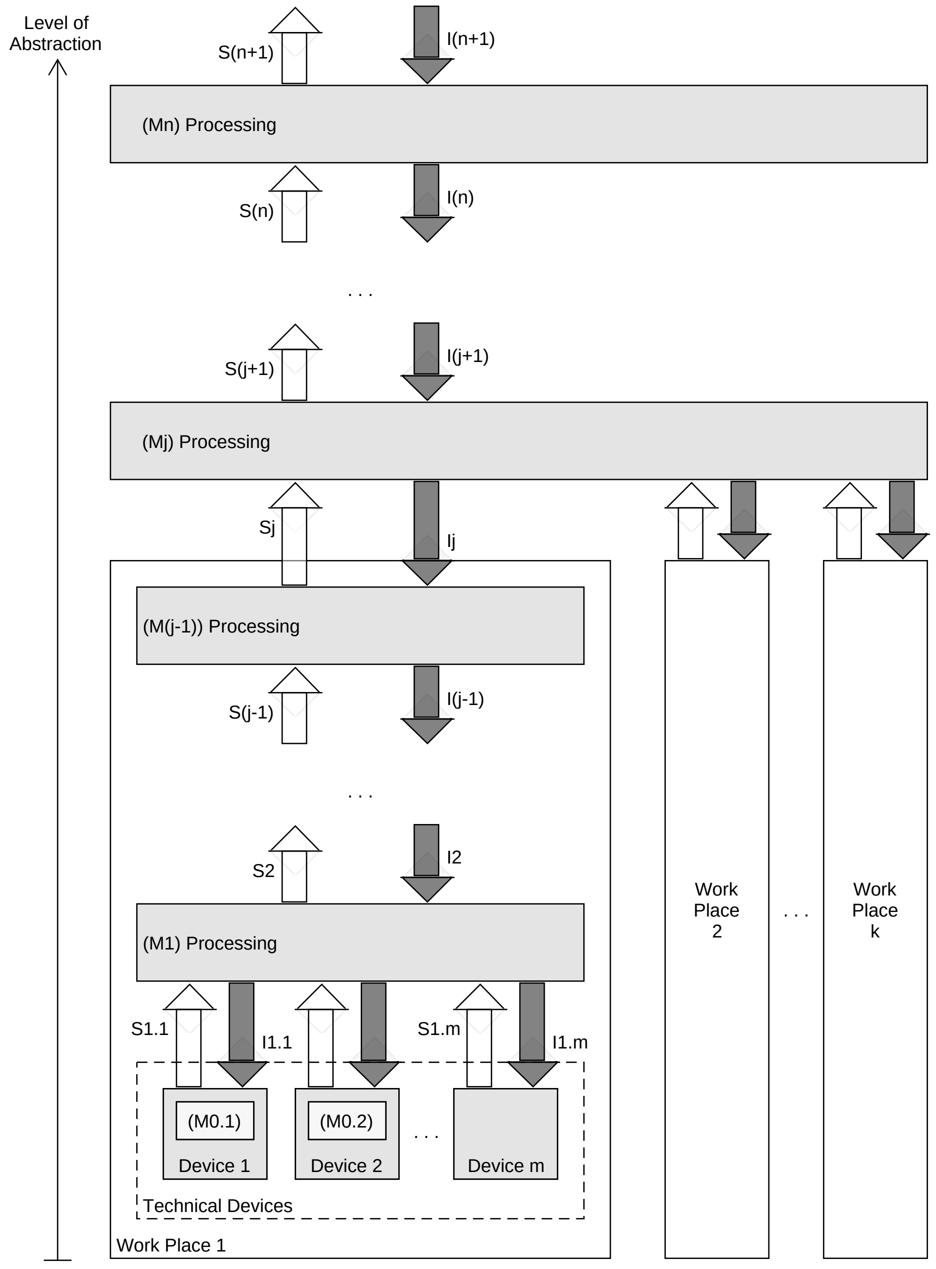

Figure 1. Multilayer architecture of cognitive systems for supporting hybrid processes in technical work places.

Thus, the processing unit generates four different kinds of output:

- $A1$: Information on the synthesized, interpreted situation, passed on as input to the next higher layer $(i+1)$, if present
- $A2$: Set of instructions that is passed down to the next lower layer $(i-1)$, if present, or that is addressed directly to the technical devices, if $i=0$
- $A3$: Newly gathered knowledge rules for the knowledge database
- $A4$: Information on all inputs, processing steps and generated outputs, as well as on all modifications that were induced in the parameters of the neural network, to be documented in the logbook

Within the processing unit, algorithmic logic and neural networks can be combined in many different topologies to build the processing unit. In particular, they can be connected in series or in parallel, or in a combination of both, involving one or more instances of both algorithmic logic and neural network.

For example, a modern thermometer, which includes both a temperature sensor and some form of algorithmic logic, could

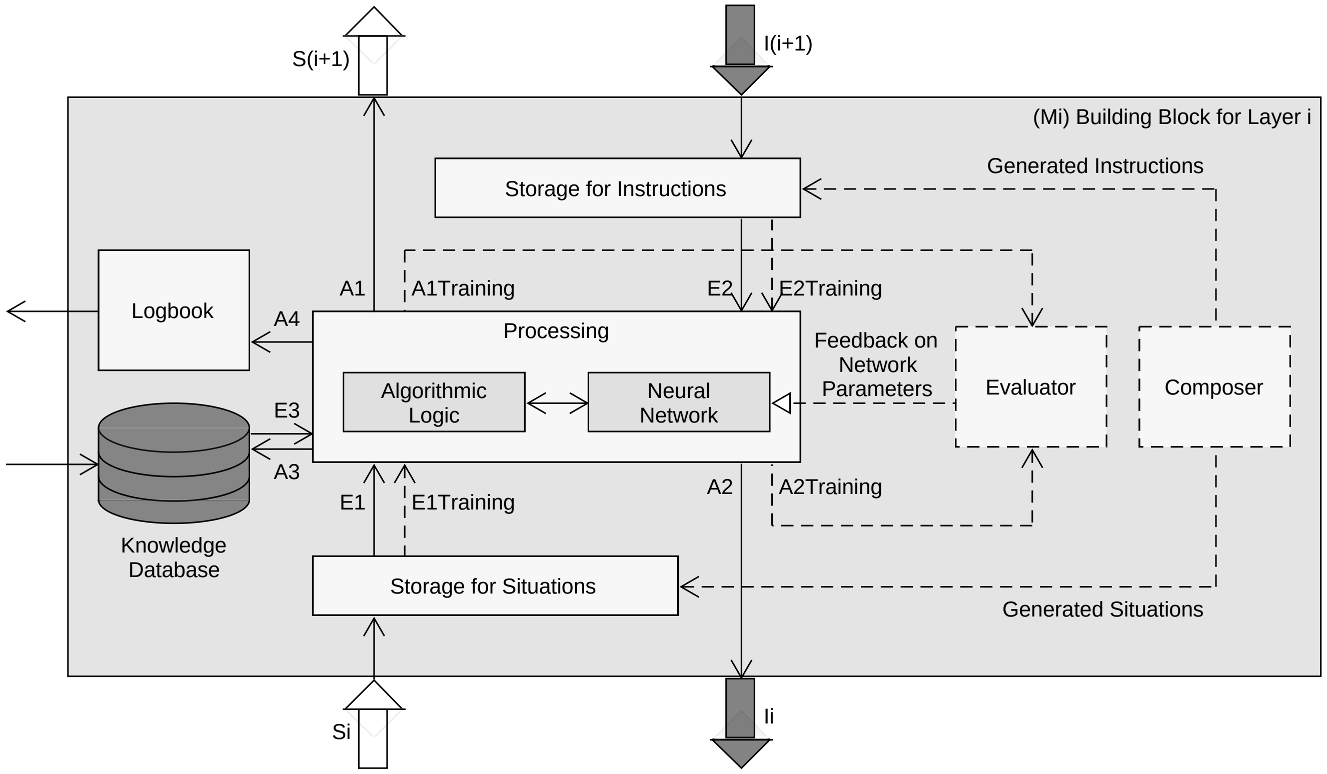

Figure 2. Pattern of building block that implements each layer.

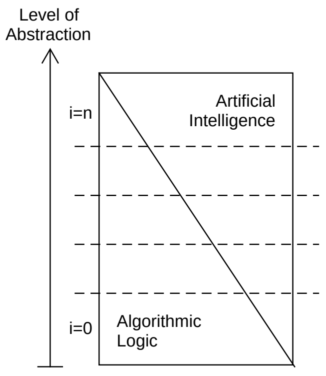

Figure 3. Varying proportions of algorithmic logic and artificial intelligence.

monitor whether the current temperature exceeds a previously defined threshold. If the threshold value is exceeded, the algorithmic logic passes the history of measured values on to the neural network, which synthesizes and analyses this data in order to identify the current situation.

Another example for a neural network connected in-series to a subsequent algorithmic logic (i.e., the other way round from the above example), would be to enter a variety of measured data from different devices into the neural network, which derives from this data the overall situation of the work place. After the neural network classified and identified the situation, this information on the situation is passed on to an algorithmic logic that executes the predefined process that deals with this type of situation.

An example for running algorithmic logic and neural network in parallel, followed by a second algorithmic logic that is connected in series, would be some sort of security mechanism, where both algorithmic logic and neural network process the same input data individually and independently of each other. After both components reached their classification result, a subsequent algorithmic logic compares the individual results and decides on further processing steps.

Note that this combination of neural network and algorithmic logic constitutes an ensemble. Generally, an ensemble consists of a set of classifiers of different types (e. g. neural network, decision tree, ...) that are trained individually. In any given situation, each classifier computes its own individual prediction. The predictive results that are delivered by the different classifiers are then combined into the prediction of the entire ensemble. More generally, by ensemble methods, we denote meta algorithms that combine several different machine learning techniques into a single predictive model, in order to reduce distortion and to improve the predictive quality. Studies show that the predictive quality achieved by an ensemble is usually more precise than each of the separate classifiers within the ensemble [11].

For training purposes, the building block provides another two components: a composer and an evaluator, depicted in Figure 2 by dashed lines. The composer generates instructions and situations and inserts them into the respective storages as training data. To achieve this, the composer can either generate this new data from scratch, or cut and paste snippets of "real" data from the storages into new sequences.

During training mode, the processing unit works on this training data $E1Training$ and $E2Training$ and processes it into the resulting answers $A1Training$ and $A2Training$, which are then passed on to the evaluator. The evaluator's resulting verdict is re-entered into the neural network, to adapt the parameters within the neural network, as necessary. The criteria that form the basis for the evaluator's assessment are specified by a set of rules, reflecting goals and basic values. For example, they can postulate the maximization of security, or the minimization of costs.

Note that composer and evaluator are active only during training of the neural network, but not during operations.

VI. Generating Training Data that Describes new Situations

By generative adversiarial networks (GANs), we denote a class of algorithms in artificial intelligence that is used for unsupervised learning. First ideas on this type of contradictory architectural patterns were developed by [12]. To achieve this, two separate neural networks are needed, which interact by exchanging data. Basically, the two neural networks involved in this approach compete in a zero-sum game [13], where one network acts as the generating model and the other network as a discriminating model [14]. While the second network, i.e., the discriminator, learns how to avoid results that are classified as *bad*, the first network (the generator) tries to challenge the second network so that it delivers a *bad* answer. Thus, over time, the evaluation of the discriminator improves step by step.

Later on, this initial idea of Li, Gauci and Gross was labeled as *Turing Learning* and applied, among others, for generating quasi-realistic photographs [15] [16] [17]. We now transfer the underlying idea into a new context: generating new types of situations that were hitherto unknown, but nevertheless make sense with respect to the laws of physics.

To achieve this, one neural network (the composer, a DCGAN) generates candidates for new situations, which are then evaluated by the other network (the evaluator, a CNN). The evaluator network is trained by presenting it with specific instances of the generated data, until it reaches a satisfactory degree of accuracy [18] when trying to identify the generated data instances from the original real ones. To achieve this, the evaluator calculates an error between the true original and the generatied data.

Both networks apply backpropagation to improve their results. As a consequence, step by step the composer learns to create situations that are more and more realistic (and adhere to the laws of physics), while the evaluator improves its ability to correctly identify situations that were synthetically generated.

Once a sufficient accuracy is reached, a thus generated situation may be inserted into the processing unit of the building blocks as regular training data. Then, the composer is again seeded with a new input from the initial data space. On this new, hitherto unknown situation compser and evaluator are trained again. Thus, synthetic situations can be generated over and over again, until no futher new situations appear.

Note that it is crucial that situations that were synthetically generated are not entered into the composer/evaluator-pair as original true data. Otherwise, the entire system is in danger of losing its touch of reality. However, it might be interesting to investigate wether in this case, the system would converge towards a single stable state, or whether the latent space shows different plateaus.

In some architectures, the evaluators directly influence the processes that synthetically generate the situation data. An analysis of the consequences of this kind of interdependency between composer and evaluator has yet to be analyzed as part of further research.

VII. Example Execution

To illustrate which kinds of tasks are dealt with on the different layers and what kind of information is processed, Figure 4 visualizes the information flow over time for an example system and a specific exemplary situation.

The physical layer of our example work place contains seven devices, six of which are directly connected to the cognitive system: a thermometer $D1$, three cameras $D2$, $D3$ and $D4$, a loudspeaker $D5$ and an e-mail system $D6$. In addition, the work place comprises a traditional heater $D7$, which is not data connected to the cognitive system, but observed by thermometer $D1$ and the three cameras.

Adhering to the generalized architecture in Figure 1, our exemplary cognitive system is structured into four processing layers: $M1$ for signal processing close to the technical devices, $M2$ for reactions which combines short signal snippets into larger situation contexts, $M3$ for drawing conclusions and $M4$ for overall organization. Note that layer $M4$ merges several work places ($WP2$ and $WP3$), in addition to the work place in focus.

In the diagram, time is discretized into time steps, progressing from top to bottom for reasons of readability. (As a consequence, processing layers are arranged vertically, with abstraction increasing from left to right.) Within each time step, all processing actions are executed in parallel. Note that the diagram in Figure 4 abstracts from the processing time that is required in each processing layer. Consider processing to take place at the transition from each time step to its successor, in parallel for each processing layer.

In time step 1, the technical devices pass the data they observed on to layer $M1$ for signal processing, as situation information. More precisely, thermometer $D1$ communicates a series of measured temperature values, each labeled with a time stamp. All three cameras continuously gather images from their respective sections of the work place. Each camera contains basic image processing facilities, which allow for identification of previously registered work place personnel. Thus, cameras $D2$ and $D3$ communicate that they identified person "Klaus" at a certain position in the work place. In contrast to this, camera $D4$ did not identify any persons in the section of the work place that it observes.

Layer $M1$ receives this situation information from the technical devices and stores it in its storage for situations. On this basis, it aggregates the gathered information and synthesizes it into a more complex understanding and larger context of a situation, thus increasing the level of abstraction. Here, layer $M1$ realizes that both cameras $D2$ and $D3$ identified the same

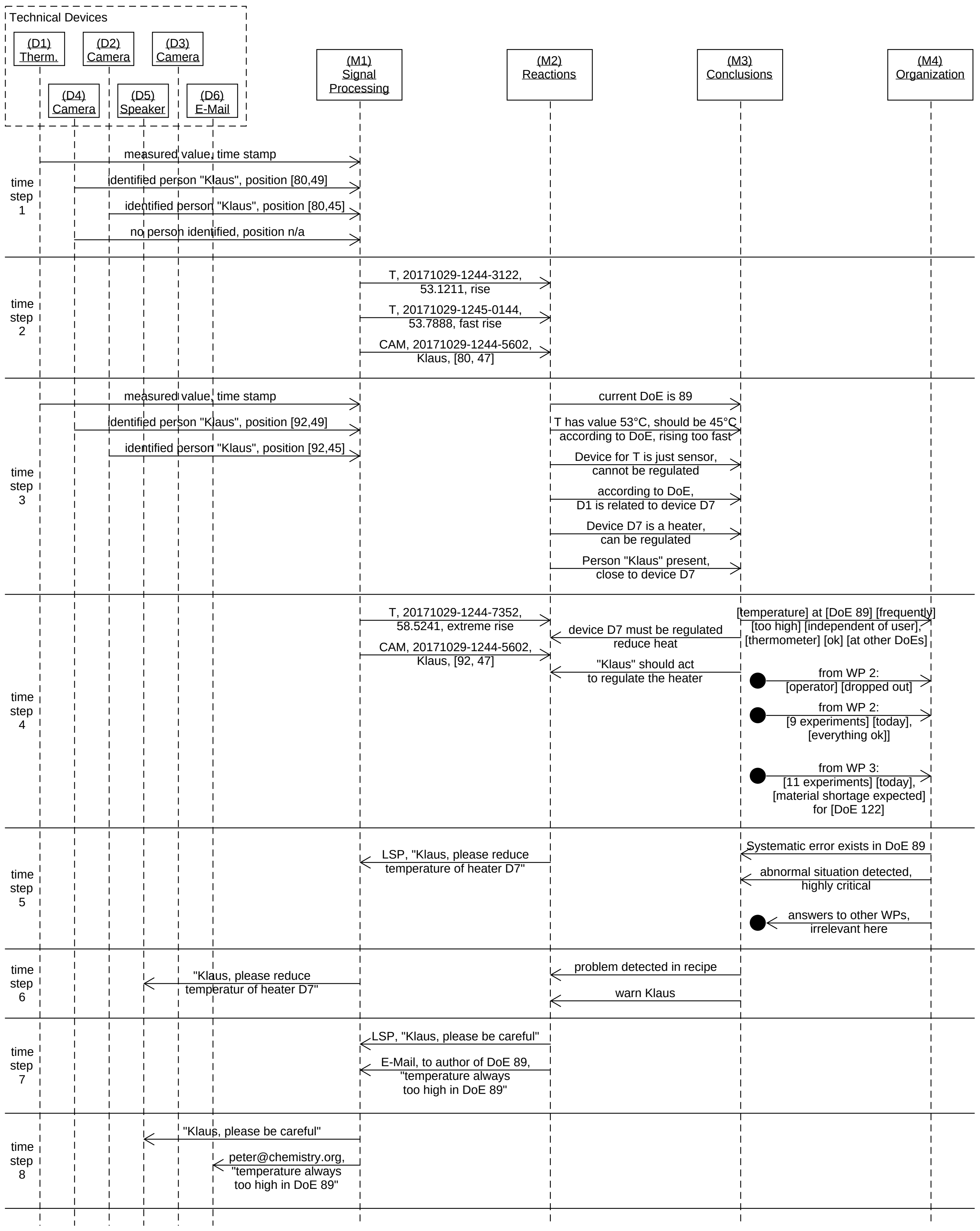
Technical Devices
(D1) Therm.
(D2) Camera
(D3) Camera
(D4) Camera
(D5) Speaker
(D6) E-Mail
(M1) Signal Processing
(M2) Reactions
(M3) Conclusions
(M4) Organization
time step 1
measured value, time stamp
identified person "Klaus", position [80,49]
identified person "Klaus", position [80,45]
no person identified, position n/a
time step 2
T, 20171029-1244-3122, 53.1211, rise
T, 20171029-1245-0144, 53.7888, fast rise
CAM, 20171029-1244-5602, Klaus, [80, 47]
time step 3
measured value, time stamp
identified person "Klaus", position [92,49]
identified person "Klaus", position [92,45]
current DoE is 89
T has value 53°C, should be 45°C according to DoE, rising too fast
Device for T is just sensor, cannot be regulated
according to DoE, D1 is related to device D7
Device D7 is a heater, can be regulated
Person "Klaus" present, close to device D7
time step 4
T, 20171029-1244-7352, 58.5241, extreme rise
CAM, 20171029-1244-5602, Klaus, [92, 47]
device D7 must be regulated reduce heat
"Klaus" should act to regulate the heater
[temperature] at [DoE 89] [frequently] [too high] [independent of user], [thermometer] [ok] [at other DoEs]
from WP 2: [operator] [dropped out]
from WP 2: [9 experiments] [today], [everything ok]]
from WP 3: [11 experiments] [today], [material shortage expected] for [DoE 122]
time step 5
LSP, "Klaus, please reduce temperature of heater D7"
Systematic error exists in DoE 89
abnormal situation detected, highly critical
answers to other WPs, irrelevant here
time step 6
"Klaus, please reduce temperatur of heater D7"
problem detected in recipe
warn Klaus
time step 7
LSP, "Klaus, please be careful"
E-Mail, to author of DoE 89, "temperature always too high in DoE 89"
time step 8
"Klaus, please be careful"
peter@chemistry.org, "temperature always too high in DoE 89"

Figure 4. Information flows between processing layers, for an exemplary execution of the system.

person, Klaus, and calculates the position of Klaus in the work place. Furthermore, layer $M1$ analyzes the sequence of temperature information measured by the thermometer. Here, layer $M1$ realizes that the temperature rises really quickly. This aggregated information is then passed on to layer $M2$ for reactions in time step 2. Information is organized according to the syntactic pattern of type of device, time stamp and two more items of structured information, whose syntax and semantics are relative to the type of the device.

The knowledge database of layer $M2$ contains information on the experiments that are carried out within the work place, referenced as DoE (i.e., design of experiments) in Figure 4. From previous context information, layer $M2$ is aware that DoE 89 is currently processed. $M2$ realizes that the measured temperature rises both faster and higher than specified in the "recipe" that is defined in DoE 89, and that the thermometer T is correlated with the heater $D7$. As well, $M2$ identifies that the thermometer T is merely a sensor and thus cannot be regulated, whereas the heater $D7$ can be regulated. Furthermore, $M2$ identifies that person Klaus is located close to the heater $D7$. All this synthesized situation information is passed on to layer $M3$ for conclusions in time step 3, and stored there in $M3$'s storage for situations. In addition, the technical devices keep sending situation information towards level $M1$ continuously. In Figure 4, this information flow is indicated as well for time step 3.

As a next step, layer $M3$ deduces from the situation information that device $D7$ is about to overheat and that Klaus is still present and able to act. Furthermore, experience gathered from previous situations indicates that in experiments that are executed according to DoE 89, temperature problems arise rather frequently, independently of the current human operator. In addition, $M3$ realizes that heater and thermometer work fine in other experiments, and thus seem to be in order technically. As a result, in time step 4 layer $M3$ communicates as instructions to level $M2$ that the heater $D7$ must be regulated, and that Klaus should act to regulate the heater. In addition, $M3$ communicates to level $M4$ for organization that temperature problems occur frequently when executing DoE 89. In addition, during time step 4 layer $M1$ passes on its newly aggregated situation information on to level $M2$, indicating that the temperature is still rising and that Klaus has moved towards the heater. As layer $M4$ for organization joins the information of several work places, additional information arrives as input from other work places during time step 4.

Based on all this situation information, layer $M4$ for organization deduces that there might exist a systematic problem in the documentation of DoE 89, such as wrong instructions. As well, $M4$ identifies that the situation in work place 1 is highly critical. Therefore, $M4$ specifies suitable instructions and passes them down to level $M3$ during time step 5. In parallel, layer $M2$ processes the newly arrived situation information in the light of the instructions that were handed down towards layer $M2$ during time step 4. As the temperature is still rising rapidly, $M2$ passes down to layer $M1$ the instruction that the loud speaker $D5$ should instruct Klaus to reduce the temperature of heater $D7$.

This instruction is translated by layer $M1$ into appropriate signals for the loud speaker $D5$, which are transferred to $D5$ during time step 6. In addition, layer $M3$ for conclusions derives from the instructions received from $M4$, in combination with the information on the ongoing situation, that a problem was detected in the recipe of DoE 89 and that Klaus has to be warned about the critical situation. Corresponding instructions are passed from $M3$ to $M2$ during time step 6.

$M2$ translates these abstract instructions into device specific instructions and passes them down to signal processing $M1$ during time step 7.

Finally, $M1$ generates the appropriate, device specific signals for the loud speaker $D5$ and for the e-mail system $D6$, respectively, and passes them down to their respective recipients, which execute them appropriately.

VIII. Prototypical Proof of Concept

A prototypical proof of concept addressing the lower layers of the example presented here was realized as a show case, using IBM Watson [19] [20] as well as Tensorflow [21] [22] for the neural network processing.

In this prototypical realization, the layer $M0$ comprises a variety of cognitive channels implemented via IoT hardware: three cameras, one projector, one microfone, speakers addressed via five Raspberry PI (based on Python), one ph-Meter, one scale, and temperature sensors addressed via three ESP8266 (based on Lua). A selection of IBM Watson services is used to realize cognitive abilities on this layer (STT speech to text, TTS text to speech, and visual recognition via REST).

Both layers $M1$ and $M2$ are realized in the cloud. Processing is based on IBM Bluemix Services [23] that are implemented in TypeScript. In addition, more complex cognitive abilities from the IBM Watson portolio are included on layer $M2$, e.g., NLC natural language classifier, which is addressed via REST as well. Furthermore, the storage for situations on layer $M2$ is realized via a NoSQL Couch database.

Layer $M3$ runs on local hardware. Processing logic is implemented in Python. Situation information is retrieved from $M2$ by download. Within the prototype, test cases were classified by hand and are processed by a convolutional neural network (based on MNIST implementation) in Tensorflow, using two convolutional / pooling layers and the dense layer with ten cases. Classification results are uploaded manually to layer $M2$.

In spite of the rather small numer of test cases, the system achieves good classification results. Note that for layer $M3$ to deliver more comprehensive classification results, a much larger amount of data would have to be collected on level $M2$, comprising at least 1000 laboratory days. Nonetheless, by this prototypical realization and the test cases under consideration, it was possible to validate the feasibility of our architecture.

Note that for $M4$, an even greater amount of data would be necessary, due to the complexity of decisions and reasoning that take place in this layer. As there are no predefined networks available for this domain, training has to be executed from scratch. Thus a large amount of "experience" in terms of data must be gathered before more meaningful results can be achieved on this layer.

IX. Critical Discussion

In principle, it would be possible to implement a cognitive assistant system with matching abilities as a monolithic block of neural networks, rather than using our multilayer architecture. However, this monolithic block would have to handle the entire complexity by itself, thus requiring an extreme amount of training that greatly exceeds what can be handled even by modern hardware. This complexity can be handled only by partitioning the system into smaller parts that are implemented and trained separately.

All in all, the structure of the building blocks ensures that the system corresponds to a sequence of symbolic components (for persistently storing information on situations and instructions) and subsymbolic, algorithmic components that process this information. Thus, the overall processing is clearly structured into distinct layers, which facilitates the implementation of processing units and allows for their individual training, as well as for the analysis of the resulting information.

Note that the layering we suggest does *not* replicate the layers of the human brain. Rather, it creates levels of abstraction that are tailored to meet the specific requirements of the technical system. As a consequence, the system's cognition, and thus, its awareness, will differ from that of a human being.

As any artificial intelligence, the system has an error margin that depends, e.g., on the noisy environment, changing illumination, the possible novelty of input sequences, as well as on the imperfection of the decision system itself. Thus, it is possible that the system misinterprets a situation.

If, for example, the system's task is to identify a person "Klaus" based on data gathered by cameras and microphones, it can indeed happen that Klaus is not recognized, or that a wrong person is recognized as "Klaus" (although it is, in fact, "Peter"). In the first case, the system is aware that something did not work properly; in the second case, it is not.

Strategies for dealing with the first error case range from *retry* (i.e., issuing instructions to present oneself to the camera again) to *comment*, thus informing the user as comprehensively as necessary that something unexpected has happened. The second case, where the system is unaware of its error, is more severe. Although it cannot be entirely avoided, its possibility can be significantly reduced by sufficient training.

Currently, the person identification process is based on the matching or peoples' looks, i.e., on rather static optical features, such as the shape of the face or the hair colour. To improve security, voice recognition could be added. In addition, it would be possible to analyse and compare typical behavioral patterns of human actors, such as their specific way to execute certain movements, e.g., the way they walk or their body pose while working. As behavioural patterns are much harder to copy than mere static looks, this could increase security. However, to be able to differentiate small nuances in movements in order to correctly identify individual people, a vast amount of additional data and training would be required.

Another possibility for increasing security would be for the system to compare the current interaction pattern of the person identified as "Klaus", who is working right now in the laboratory, with history data on previous interactions between Klaus and the system. If the system realizes that Klaus acts and reacts differently than usual, e.g., shows different response times, or executes the different steps in a faster or slower way, it could issue a warning to some other control unit (human or otherwise), indicating the possibility that the person in the laboratory might not be "Klaus" after all; or that it is indeed Klaus, but Klaus on a bad day (i.e., headachy or preoccupied), and thus not acting up to his usual well-focused, competent self. Both cases would require some action to reinstate the overall system's security.

Note that layer $M2$, which is responsible for realizing rather basic reactions, incorporates several algorithmic mechanisms for identifying, and then blocking, discrepancies to usual patterns, contradictions to what is expected and desired, maloperation or even blatant misuse. In contrast to this, incorrect decisions on layer $M4$, in our example responsible for the overall organization, are much harder to tackle. Similar to decision processes in biological systems, they can only be tackled by increased training and subsequent debating [24].

In each building block, processing is performed by a combination of neural networks and algorithmic logic, which are integrated into an ensemble. Note that the proportions of the different parts varies, depending on the level of abstraction of the layer that is realized by the building block. Thus, each layer can incorporate different componentes that are self-contained and pre-trained individually. Examples for these components are a device for identifying laboratory glassware, a security check or a process control logic.

We expect that in future versions of our system, each processing unit will be manually composed from a set of individual modules, neural networks or algorithms, as suggested by [25] [26]. This development is supported by the increasing possibilities for acquiring modules that are pre-trained for certain tasks. Platforms such as Model Asset eXchange (MAX) from IBM or algorithmia facilitate the trading with pre-trained models.

X. Conclusion and Future Work

We introduced a multilayer architecture for cognitive systems that support the operation of technical work places, in which hybrid processes (partially executed manually, and partially using technical devices) are executed. To ensure efficiency and adaptability, we structured this architecture into separate layers on different levels of abstraction. Each layer deals with specific kinds of tasks and processes the corresponding kind of information, which again is organized into different levels of abstraction.

Each layer of the conceptual architecture is realized by a building block, which incorporates aspects of both algorithmic logic and artificial intelligence. We provided a template defining the glass box view of these building blocks. Based on this template and the conceptual architecture, it is possible to develop cognitive systems that scale appropriately, to meet the demands of the application context under consideration.

To ensure adequate training of the different processing units, each building block incorporates generative adversarial networks that cooperate as composer and evaluator, in order to generate training data that exceeds situations that have been physically measured in the past.

As a next step, we demonstrated the interaction and cooperation of the different layers for a concrete example, specified from the context of a chemical laboratory. Furthermore, we sketched a prototypical proof of concept that addresses the lower layers of the presented example. This prototype was run on a small number of test cases, to validate the feasibility of our architecture.

For extending the prototypical system towards a more comprehensive classification of situations and recommendations of instructions, extensive laboratory data will have to be collected, as a basis for properly training the cognitive system.

Currently, introducing an assistant system that is based on the architecture presented here, into the workplaces of a large German manufacturer of spectacles and glasses is under discussion [27]. Generally, our system architecture is best suited for supporting workplaces of a highly technical and scientific character, such as the premises of optometrists or hearing aid acousticians, as well as for laboratory environments in a chemical, pharmaceutical or medicinical context.

Parts of this work are closely related to an innovation that is covered by the German patent application 10 2017 126 457.4.

References

[1] V. Thurner and T. Gressling, "A Multilayer Architecture for Cognitive Systems – Supporting well-defined processes that are partially executed manually in technical work places," in Cognitive 2018 – The Tenth International Conference on Advanced Cognitive Technologies and Applications, Barcelona, Spain. IARIA, 2018, pp. 63–71.

[2] M. Burgin and G. Dodig-Crnkovic, "A Taxonomy of Computation and Information Architecture," in ECSAW, Dubrovnik, Croatia. ACM, 2015, pp. 7:1–7:8, DOI: 10.1145/2797433.2797440.

[3] C. Leibold and M. Spies, "Towards a Pattern Language for Cognitive Systems Integration," in EuroPLoP, Irsee, Germany. ACM, 2014, pp. 17:1–17:9, DOI: 10.1145/2721956.2721968.

[4] M. Aehnelt and B. Urban, "The knowledge gap: Providing situation-aware information assistance on the shop floor," in HCI, Los Angeles, USA. Springer, 2015, pp. 232–243, DOI: 10.1007/978-3-319-20895-4_22.

[5] O. Korn, M. Funk, and A. Schmidt, Assistive systems for the workplace. IGI Global, 2015, pp. 121–135, DOI: 10.4018/978-1-4666-8200-9.ch097.

[6] T. Kosch, Y. Abdelrahman, M. Funk, and A. Schmidt, "One size does not fit all – Challenges of providing interactive worker assistance in industrial settings," in UbiComp/ISWC, Maui, USA. ACM, 2017, pp. 1006–1011, DOI: 10.1145/3123024.3124395.

[7] A. Srivastava and P. Yammiyavar, "Design of multimodal instructional tutoring agents using augmented reality and smart learning objects," in ICMI, Tokyo, Japan. ACM, 2016, pp. 421–422, DOI: 10.1145/2993148.2998531.

[8] M. Funk, T. Kosch, and A. Schmidt, "Interactive worker assistance: Comparing the effects of in-situ projection, head-mounted displays, tablet, and paper instructions," in UbiComp. ACM, 2016, pp. 934–939, DOI: 10.1145/2971648.2971706.

[9] S. Büttner, O. Sand, and C. Rocker, "Exploring design opportunities for intelligent worker assistance: a new approach using projetion-based AR and a novel hand-tracking algorithm," in AmI, Malaga, Spain. Springer, 2017, pp. 33–45, DOI: 10.1007/978-3-319-56997-0_3.

[10] J. Anderson, M. Matessa, and C. Lebiere, "ACT-R: A Theory of Higher Level Cognition and its Relation to Visual Attention," Human–Computer Interaction, 1997.

[11] Z.-H. Zhou, J. Wu, and W. Tang, "Ensembling Neural Networks: Many could be better than All," Artificial Intelligence, vol. 137, 2002, pp. 239–263.

[12] J. Schmidhuber, "Learning Factorial Codes by Predictability Minimization," Neural Computation, vol. 4, no. 6, 1992, pp. 863–879.

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and J. Bengio, "Generative Adversarial Networks," arXiv 1406.2661, 2014.

[14] W. Li, M. Gauci, and R. Gross, "A Coevolutionary Approach to Learn Animal Behavior Through Controlled Interaction," in GECCO, Amsterdam, Netherlands, 2013, pp. 223–230.

[15] C. e. a. Ledig, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," arXiv [cs.CV], 2016.

[16] S. e. a. Reed, "Generative Adversarial Text to Image Synthesis," arXiv [cs.NE], 2016.

[17] H. e. a. Zhang, "Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks," arXiv [cs.CV], 2016.

[18] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation," arXiv [cs.NE], 2017.

[19] K. Mak, H. Pilles, M. Bertl, and J. Klerx, "Wissensentwicklung mit IBM Watson in der Zentraldokumentation (ZentDok) der Landesverteidigungsakademie," https://www.academia.edu/37674334/Wissensentwicklung_mit_IBM_Watson, accessed 11/2018, Landesverteidigungsakademie, Tech. Rep.

[20] Y. Saxena, "IBM Watson: Determining the presence of an Artificially Intelligent Nature," https://www.academia.edu/23441941/IBM_Watson_Determining_the_presence_of_an_Artificially_Intelligent_Nature, accessed 11/2018.

[21] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16). Savannah, GA: USENIX Association, 2016, pp. 265–283. [Online]. Available: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi

[22] F. Ertam and G. Aydın, "Data classification with deep learning using Tensorflow," in 2017 International Conference on Computer Science and Engineering (UBMK), Oct 2017, pp. 755–758.

[23] J. Moore, M. Hirzalla, R. Osowski, S. Chowdhury, and V. Gucer, "IBM Bluemix Architecture Series: Web Application Hosting on Java Liberty – Leveraging best practice and reference architectures for cloud," http://www.redbooks.ibm.com/redpapers/pdfs/redp5184.pdf, accessed 11/2018, IBM, Tech. Rep.

[24] IBM Research, "IBM Project Debater – Can artificial intelligence expand a human mind?" https://www.research.ibm.com/artificial-intelligence/project-debater/, accessed 11/2018.

[25] C. Shu and D. H. Burn, "Artificial Neural Network Ensembles and Their Application in Pooled Flood Frequency Analysis," Water Resources Research, vol. 40, 2004, p. 655.

[26] X. Yao and M. M. Islam, "Evolving Artificial Neural Network Ensembles," IEEE Comput. Intell. Mag., vol. 3, 2008, pp. 31–42.

[27] ARS Computer und Consulting GmbH, "ARS Beratung – Architektur und Cognitive Services – Rodenstock GmbH," https://web.ars.de/wp-content/uploads/2018/04/ARS_Referenz_Rodenstock_Machbarkeitsstudie_Cognitive_AI.pdf, accessed 11/2018.