Computational Argumentation & Cognitive AI

Emmanuelle Dietz[1](✉), Antonis Kakas[2], and Loizos Michael[3,4]

[1] TU Dresden/Airbus Central R&T, Dresden, Germany
emmanuelle.dietz@airbus.com
[2] University of Cyprus, Nicosia, Cyprus
[3] Open University of Cyprus, Latsia, Cyprus
[4] CYENS Center of Excellence, Nicosia, Cyprus

Abstract. This tutorial examines the role of Computational Argumentation at the theoretical and practical level of Human-centric AI. It rests on the central role that argumentation has in human cognition rendering argumentation as a possible foundation for the two basic elements of intelligence, namely learning and reasoning, in a way that is suitable for human-centric AI. The tutorial examines argumentation as a basis for cognitive technologies of Learning and Explainable Inference or Decision Making and their application in today's AI.

Keywords: Argumentation · Human-centric AI · Learning and reasoning in argumentation · Cognitive modeling

1 Introduction

This tutorial follows a first tutorial, "Argumentation in AI" on Abstract Argumentation in AI, given at the same ACAI 2021 school. The purpose of the current tutorial is to connect argumentation with Human-centric AI by examining the natural link of argumentation with human cognition and the two basic elements of intelligence, learning and reasoning. The main **learning objective** of the tutorial is for participants to appreciate the potential central role of argumentation for Human-centric AI and how this can form the basis for developing real-life applications. The tutorial is structured into four parts, as follows:

- **Section 2: Structured Argumentation**, presenting a general review of structured argumentation as the underlying framework on which applications of (explainable) Human-centric AI can be build. These general ideas are illustrated within the concrete structured argumentation framework of *Gorgias* and its associated system, available at **Cloud Gorgias**.[1]

[1] http://gorgiasb.tuc.gr/GorgiasCloud.html.

Tutorial at the HUMANE-AI NET advanced course 2021 on Human Centered AI.

M. Chetouani et al. (Eds.): ACAI 2021, LNAI 13500, pp. 363–388, 2023.
https://doi.org/10.1007/978-3-031-24349-3_19

- **Section 3: Cognitive Argumentation**, examining the natural link between human reasoning and argumentation. The COGNICA system[2] implements such a Cognitive Argumentation framework. Also the link of argumentation to existing Cognitive Architectures, such as ACT-R, is examined.
- **Section 4: Argumentation for Learning**, introducing the features of argumentation that make it a fitting target language for learning and explanations. It showcases this natural fit by presenting two protocols that learn and represent knowledge in the language of argumentation.
- **Section 5: Real-life Applications of Argumentation**, presenting an argumentation-based software development methodology for acquiring the knowledge required for building systems under a general "mind-like" architecture. This methodology is illustrated through a series of real-life application systems and the major challenges it poses.

In the tutorial repository[3] one can find further details on all parts of the tutorial, e.g., extended presentations of examples or other illustrative applications. Note also that for each section of this tutorial, a general bibliography is listed separately at the end, without explicit citations in the text. For a more complete bibliography the reader can consult the tutorial repository.

2 Structured Argumentation

In contrast to Abstract Argumentation, **Structured Argumentation** puts the emphasis in providing argumentation frameworks that can be used to model and develop applications of argumentation. They provide the necessary scaffolding for dialectic argumentative reasoning (or inference) to be mapped into, and applications to be build on top of this.

At a general and abstract level a structured argumentation framework consists of a triple $\langle \mathcal{A}rgs, \mathcal{A}tt, \mathcal{D}ef \rangle$ where $\mathcal{A}rgs$ is a set of arguments, $\mathcal{A}tt$ an attack relation between arguments and $\mathcal{D}ef$ a defense relation between arguments. Typically, the defense relation $\mathcal{D}ef$ is a subset of the attack relation $\mathcal{A}tt$ and relates to the relative strength between arguments.[4] Informally, $(a, b) \in \mathcal{D}ef$ means that argument a is at least as strong as b, and can thus provide a defense against b.

In Structured Argumentation, like in abstract argumentation, we can give an underlying dialectical semantics for the acceptability of arguments. For example, a subset of arguments Δ **is admissible** iff (a) it is not self-attacking, i.e., there are no arguments a, b in Δ such that $(a, b) \in \mathcal{A}tt$ and (b) for any counter-argument c against Δ, i.e., $(c, a) \in \mathcal{A}tt$ holds for some argument a in Δ, Δ defends against c, i.e., $(d, c) \in \mathcal{D}ef$ for some d in Δ. This then maps directly into a dialectic process of inference of recursively considering attacks against an argument supporting a desired conclusion and defending against these attacks

[2] http://cognica.cs.ucy.ac.cy/COGNICAb/index.php.

[3] https://cognition.ouc.ac.cy/argument.

[4] Alternatively, the notion or terminology of a *defeating attack* is used instead to express that an attack is strong enough to defeat the argument that it is attacking.

with possibly the help of other arguments thus building an admissible Δ. We call such an admissible set Δ a **case** for the inferred conclusion.

In practice, structured argumentation frameworks are realized in an application domain via triples of $\langle \mathcal{A}s, \mathcal{C}, \succ \rangle$ where $\mathcal{A}s$ is a set of (parameterized) argument schemes, instances of which form arguments, $\mathcal{C}$ is a conflict relation between the argument schemes and the arguments constructed from these schemes and $\succ$ is a priority or preference relation again between the argument schemes and their arguments. **Argument schemes** are parameterized named statements of association $\mathsf{AS} = (\mathsf{Premises} \rhd \mathsf{Position})$ between some information called $\mathsf{Premises}$ and another statement called the $\mathsf{Position}$ or Claim.

The conflict relation $\mathcal{C}$ is typically defined through the language of the application domain, e.g., through some global notion of incompatibility between statements in the language, possibly also augmented with a direct expression of conflict between two argument schemes and/or particular instances of these. Given such a conflict relation we can build the attack relation between arguments by identifying three different types of attacks, called **rebuttal, undermining or undercutting attacks**. The first type results when the claim of the attacking argument conflicts with the claim of the argument attacked, the second type when it conflicts with a premise of the argument attacked and the third type when the two arguments have been declared as conflicting — the conflict is on the link of the argument that it is attacked.

Example 1. Consider argument $arg_1 : Arrival\_of\_Ambulance \rhd Pick\_up\_Patient$, i.e., the arrival of an ambulance supports the claim that it will pick up a patient (from the place of arrival). A rebuttal attack against this is given by the argument $arg_2 : No\_Ambulance\_Siren \rhd Not\_Pick\_up\_Patient$, i.e., the argument supporting the opposite claim when there is no ambulance (arriving) with its siren on, whereas the argument $arg_3 : Broken\_Ambulance \rhd Not\_Arrival\_of\_Ambulance$ supporting the claim that an ambulance cannot arrive based on the premise that it is broken is an undermining attack on arg_1. Finally, the argument $arg_4 : Arrival\_of\_Ambulance \rhd Pick\_up\_Nurse$ is an undercutting attack against arg_1 as this goes against the actual link of the argument: arg_1 claims that the reason it has arrived is to pick up a patient whereas arg_4 claims it is to pick up a nurse.

The third component, the priority or strength relation $\succ$ between arguments, is used to build the defense relation of an application argumentation framework. Informally, in most frameworks arg_1 defends against arg_2 iff arg_1 conflicts with arg_2 and arg_1 is not of lower priority than arg_2, i.e., $arg_1 \not\prec arg_2$. In contrast to the conflict relation which is static the priority relation is not so, but can be highly *context-sensitive* depending crucially on (how we perceive) the current state of the application environment.

To illustrate the process of how an argumentation framework is built dynamically through a changing current environment, let us consider the following example from the domain of common-sense temporal reasoning.

Example 2. Suppose we read the following piece of text: "Bob came home and found the house in darkness. He turned on the light switch in the hall." Consider the question "Is the hall still in darkness?". Can we explain[5] how (most) people reach the conclusion or explain why "the hall now is illuminated"?

One way to do this within an argumentation perspective is as follows.

- From the information *Room_in_darkness_at_T* using the general argument schema that properties *persist* in time we have the argument $arg_1 : \{Room\_in\_darkness\_at\_T \rhd Room\_in\_darkness\_at\_T^+\}$ supporting the claim that the hall is still in darkness at some time T^+ after T.
- From the information *Turn_on_switch_at_T* using the common sense knowledge that turning on the light switch *causes* the light to come on, we have the argument: $arg_2 : \{Turn\_on\_switch\_at\_T \rhd Room\_illuminated\_at\_T^+\}$ supporting the claim that the hall is illuminated at T^+.

These two arguments are counter-arguments of each other as their claims are in conflict: in our common-sense language a room in darkness is the opposite of a room illuminated and vice-versa. Furthermore, in our common sense temporal reasoning we consider causal information stronger than the persistence of properties (when the causal action occurs at least as late as the time of the observed property that we are persisting from into the future). This gives a priority or strength to causal arguments over persistence arguments and hence to arg_2 over arg_1, i.e., $arg_2 \succ arg_1$. This in turn means that arg_2 can defend against arg_1 but not vice versa.

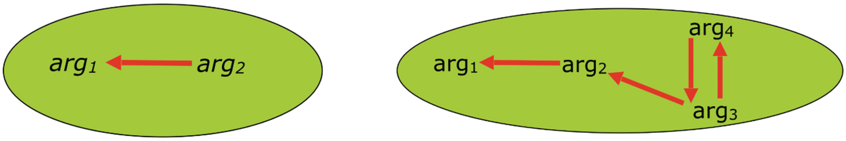


Fig. 1. The argumentation arena for the two narratives of Example 2.

We thus have an argumentation arena depicted by the left part of Fig. 1. In this figure, we have equated the defence relation with the attack relation so that we only show the non-weak attacks. From this we can see that $\{arg_2\}$ is an acceptable/admissible set of arguments forming a case supporting the conclusion *Room_illuminated_at_*T^+ and that there is no case supporting the opposite conclusion as $\{arg_1\}$ is not acceptable/admissible. Hence we are "confident" about deriving the inference that the room is not in darkness after Bob has turned on the light switch.

[5] We are not looking here for an explanation of the subconscious operation of the brain to reach this conclusion, but for an explanation at a high cognitive level that would also be helpful to some other process that would act on our conclusion.

Example 3. Consider now a more complex case by changing slightly the narrative: "The power cut had turned the house in darkness. Bob came home and turned on the light switch in the hall." Now some people may not feel confident that the hall will be illuminated after turning on the light switch. This can be attributed to two things: (1) the text now alerts us to the fact that electricity is needed for the light to come on, and (2) it is not clear if the power cut has ended before or after Bob came home. From an argumentation point of view new arguments come into the arena:

- From the statement about a power cut we can build the following argument: $arg_3 : \{Power\_cut\_at\_T \rhd No\_electricity\_at\_T\}$ which conflicts with argument arg_2. This is an undercutting[6] attack against arg_2 and it is, according to our common-sense knowledge, a stronger argument than arg_2. Hence arg_3 cannot be defended back by arg_2, or arg_3 is a defeating attack against arg_2. But to enable arg_3 we need to have an argument supporting its premise. We can thus extend argument arg_3 to: $\{\mathsf{hyp}(Power\_cut\_at\_T)); Power\_cut\_at\_T \rhd No\_electricity\_at\_T\}$ where $\mathsf{hyp}(Power\_cut\_at\_T)$ is a hypothetical argument supporting that the power cut holds at the time of turning on the switch. This then means that now arg_3 can be attacked by the opposite hypothetical argument supporting that the power cut did not last until time T, i.e., we have a fourth argument in the arena: $arg_4 : \{\mathsf{hyp}(No\_power\_cut\_at\_T)\}$. This argument is in conflict with arg_3 on its weak premise and thus forms an (undermining) attack on it. Importantly, it is non-comparable in strength with arg_3. Hence arg_3 and arg_4 attack and defend against each other.

Given the above we now have a new argumentation arena depicted by the right part of Fig. 1. From this we now have two acceptable/admissible subsets of arguments: $\{arg_2, arg_4\}$ forming a case supporting $Room\_illuminated\_at\_T^+$ and the case of $\{arg_1, arg_3\}$ for the opposite conclusion of $Room\_in\_darkness\_at\_T^+$. We have a **dilemma** and hence we cannot be sure either way that the room is in darkness or not after turning on the switch. This then reflects the variability in answers given by different people (see more on this in Sect. 3).

There are several Structured Argumentation frameworks in the literature and although these may appear different they share a very similar theoretical underpinning. One of the earliest such frameworks is that of the GORGIAS framework, named after the ancient Greek philosopher of dialectics, on which we will concentrate.

2.1 The GORGIAS Argumentation Framework

GORGIAS is a structured argumentation framework where arguments are constructed using a basic (content independent) scheme of **argument rules**,

[6] Indeed, this attacks the link of arg_2 not its claim or premises. There is no general conflict between $No\_electricity\_at\_T$ and $Room\_illuminated\_at\_T$ as the room can be illuminated in other ways.

denoted by Premises $\triangleright$ Claim. The Premises and Claim are literals in the syntax of Extended Logic Programming, but where negation as failure is excluded from the language[7]. An important element of the GORGIAS framework is that it allows a special class of argument rules, called **priority argument rules** that are used to express a context-sensitive relative strength between (other) argument rules. They have the same syntactic form as argument rules, but now the Claim is of a special type, $a_1 > a_2$, where a_1 and a_2 are (the names of) any two other individual argument rules. When the claim of an argument rule is not a priority statement, i.e., it is a literal in the language, this is called an **object-level** argument rule.

The purpose of priority arguments, constructed from priority argument rules, is to provide the defense relation between arguments. They are combined with other (e.g., object-level) arguments to give them strength. A **composite argument** in the framework is then a (minimal and closed) set of (instantiated) argument rules, $\Delta = (A_1, A_P)$, where, A_1, is a subset of object level argument rules and A_P is a subset of priority argument rules, referring to the other arguments in Δ. Then, informally, a composite argument, Δ_1, defends against another composite argument, Δ_2, whenever they are in conflict, and the arguments in Δ_1 are rendered by the priority arguments that it contains at least as strong as the arguments contained in Δ_2.

The GORGIAS System. The GORGIAS system allows us to code argumentation theories of the form described above and subsequently query the system to find out if there is an admissible (composite) argument that supports the query. GORGIAS has been publicly available since 2003 and has been used by several research groups to develop prototype real-life applications of argumentation in a variety of application domains. Today the GORGIAS system is available as a service over the internet in **Cloud Gorgias** at http://gorgiasb.tuc.gr/GorgiasCloud.html.

Let us illustrate the GORGIAS argumentation framework and the dialectic computational model of the GORGIAS system through a simple example. This is written below in the internal GORGIAS system language.[8] This language is build on top of Prolog where an argument rule has the form:

$$\mathit{rule}(\mathit{arg\_name}, \mathsf{Claim}, \mathit{defeasible\_premises}]) : -\mathit{non\_defeasible\_premises}.$$

arg_name is a Prolog term with which we name the arguments expressed by this rule, *non_defeasible_premises* can be any conjunction of Prolog conditions and are executed under ??? and *defeasible_premises* are conjunctions of literals executed under GORGIAS using argument rules relating to them. Priority argument rules have exactly the same form, but now Claim is *prefer*($\mathit{arg\_name}_1$, $\mathit{arg\_name}_2$) where $\mathit{arg\_name}_1$ and $\mathit{arg\_name}_2$ name two other different argument rules.

[7] Initially, the framework of GORGIAS had the name *LPwNF* : Logic Programming without Negation as failure.

[8] As we will see in Sect. 5 of the tutorial, it is not necessary to work at this internal level of GORGIAS when developing applications.

Example 4 (Commonsense Reasoning). The following argument rules express a common sense knowledge about birds (b), in particular penguins (p), flying (f) or not. We assume that we have sensors that can recognize clearly objects that are birds. They are unable to recognize directly penguins, but instead can recognize if an object walks like a penguin, how tall it is, and how far away it is.

$$
\begin{array}{l}
rule(r_1(X), f(X), []) : -b(X).\\
rule(r_3(X), p(X), []) : -walks\_like\_p(X).\\
rule(r_2(X), neg(f(X)), [p(X)]).\\
rule(r_4(X) neg(p(X)), []) : -over\_a\_meter(X).\\
rule(p_1(X), prefer(r_2(X), r_1(X)), []).\\
rule(p_2(X), prefer(r_4(X), r_3(X)), []) : -1m\_dist.
\end{array}
$$

Suppose our sensors have given us the following trusted information about a particular object with identifier obj_1: $b(obj_1), walks\_like\_p(obj_1), over\_a\_meter(obj_1)$. Can we infer that obj_1 (possibly) flies or not, i.e., can $f(obj_1)$ or $neg(f(obj_1))$ be supported by admissible arguments or not?

GORGIAS will try to build a (composite) argument Δ supporting $f(obj_1)$ starting with the argument rule $r_1(obj_1)$ which supports $f(obj_1)$ based on the premise of $b(obj_1)$. This is attacked by the argument $A = \{r_2(obj_1), r_3(obj_1)\}$ on the claim of $f(obj_1)$ of Δ. Δ itself forms a defense against this as they are equally strong. But this attacking argument can by strengthened by including in it the priority argument $p_1(obj_1)$. Now Δ as it currently stands cannot defend against this strengthened composite attacking argument. It therefore needs to look for other arguments to help it do so, and so it adds in Δ the argument $r_4(obj_1)$. This is in conflict with the attack A on the claim of $p(obj_1)$ and (in the absence of any information of how close we are to the object) these conflicting arguments of A and $r_4(obj_1)$ are of non-comparable (or equal) strength and so the latter can form a defense against the former. Thus the extended $\Delta = \{r_1(obj_1), r_4(obj_1)\}$ forms an admissible argument supporting $f(obj_1)$. Note that $A = \{r_2(obj_1), r_3(obj_1)\}$ supporting $neg(f(obj_1))$ is also admissible.

Suppose now that we also have that $1m\_dist$ holds. When we are looking for a defense against the counter-argument A, GORGIAS can now use a stronger (than above) defense by including also the priority argument $p_2(obj_1)$ resulting in a final $\Delta = \{r_1(obj_1), r_4(obj_1), p_2(obj_1)\}$. In addition, now we cannot build an admissible argument supporting $neg(f(obj_1))$. Argument $A = \{r_2(obj_1), r_3(obj_1)\}$ is attacked strongly (i.e. it cannot defend back at this) by $\{r_4(obj_1), p_2(obj_1)\}$ and there is no other argument strong enough to defend against this.

An important feature of the GORGIAS generated admissible composite argument Δ supporting a claim is that this serves as an **explanation** for the possible adoption of the claim. This explanation at the internal level of the GORGIAS framework can be naturally translated into an **application level explanation** exhibiting the desired characteristics of being **attributive, contrastive and actionable** as follows:.

- **Attributive:** Extracted from the object-level argument rules in Δ.
- **Contrastive:** Extracted from the priority argument rules in Δ.
- **Actionable:** Extracted from the hypothetical[9] arguments in Δ.

From the internal GORGIAS explanation of $\Delta = \{r_1(obj_1), r_4(obj_1), p_2(obj_1)\}$ of Example 4 we automatically generate the application level explanation:

- The statement "$f(obj_1)$" is supported by: — $b(obj_1)$ and $neg(p(obj_1))$.
- This support is strengthened: — (against $p(obj_1)$)) by: "*1m_dist*."

3 Cognitive Argumentation

In what follows, the natural link between human reasoning and argumentation will be exposed. It will present how *cognitive principles* drawn from Cognitive Psychology, Social Sciences and Philosophy can help develop an argumentation framework, called *Cognitive Argumentation*, as a case of structured argumentation, $\langle \mathcal{A}s, \mathcal{C}, \succ \rangle$, that is customized according to these cognitive principles. These principles would help us capture the context sensitive and adaptive nature of human reasoning as well as other computational features such as the "on demand" or "lazy process" of human reasoning. The framework of Cognitive Argumentation will be illustrated by discussing in detail the particular case of the *suppression task* as studied in Cognitive Psychology to understand the nature of human reasoning.

3.1 The Suppression Task

In the psychological study of the *suppression task* three groups of participants were asked to derive conclusions given variations of a set of premises. Group I was given the following two premises: *If she has an essay to finish, then she will study late in the library.* ($e \leadsto \ell$). *She has an essay to finish.* (e). The participants were asked what **necessarily** follows from the above two premises. They could choose between the following three answers: *She will study late in the library.* (ℓ) *She will not study late in the library.* ($\overline{\ell}$) and *She may or may not study late in the library.* (ℓ or $\overline{\ell}$) In group I, 96% of the participants concluded: *She will study late in the library.*

In addition to the above two premises for Group I, Group II was given the following premise: *If she has a textbook to read, then she will study late in the library.* ($t \leadsto \ell$) Still, 96% of the participants concluded that *She will study late in the library.* Finally, Group III received, together with the two premises of Group I, additionally the following premise: *If the library stays open, then she will study late in the library.* ($o \leadsto \ell$) In this group only 38% concluded that *She will study late in the library*: The conclusion drawn in the previous groups was *suppressed* in Group III.

[9] These are arguments whose premises are empty but are generally weaker than any conflicting argument grounded on some given premises.

Cognitive Principles. Humans make assumptions while reasoning, many of which are not necessarily valid under formal (classical) logic. Yet, humans are pretty good in explaining plausibly why they make these assumptions. Let us consider some such (typically) non-formal or extra-logical properties and formalize them as cognitive principles.

According to Grice, human communicate according to the *maxim of quality*, implying that humans try to be truthful. Applied to the suppression task this implies the following: When the experimenter states *She has an essay to finish*, then participants believe this information to be true. To reflect this principle, we establish (strong) factual argument schemes. Further, following Grice's *maxim of relevance*, mentioned information is assumed to be relevant. Even though mentioned information is not necessarily factual (e.g., *if the library stays open*), humans can still construct various context-dependent hypotheses supporting statements concerning this information. For this purpose we establish (weak) hypothesis argument schemes.

Consider again the conditional $(e \rightsquigarrow \ell)$: *She has an essay to finish* is sufficient support for *She will study late in the library.* Thus we say that e in $(e \rightsquigarrow \ell)$ is a sufficient condition. Similarly, t is a sufficient condition in $(t \rightsquigarrow \ell)$ Yet, *the library stays open* is not sufficient support for *She will study late in the library* in conditional $(o \rightsquigarrow \ell)$. However, *the library is not open* plausibly explains *She will not study late in the library.* Here, o in $(o \rightsquigarrow \ell)$ is a necessary condition. Conditionals with sufficient condition and conditionals with necessary condition will be denoted by $\overset{s}{\rightsquigarrow}$ and $\overset{n}{\rightsquigarrow}$, respectively. Further, we establish two types of argument schemes for both types of conditionals.

The following cognitively motivated relative strength relation among schemes will apply for the dialectic argumentation process: Fact schemes are the strongest schemes, whereas hypotheses schemes are the weakest schemes, and necessary schemes are stronger than sufficient schemes.

The Suppression Task in Argumentation. Given the above principles we can build an argumentation framework, $\langle \mathcal{A}s, \mathcal{C}, \succ \rangle$, where $\mathcal{A}s$ contains argument schemes drawn from the cognitive principles. To do so we assume that we have a cognitive state $\mathcal{S} = \langle \mathcal{F}, \mathcal{A} \rangle$ where $\mathcal{F}$ is the set of facts, and $\mathcal{A}$ is the set of relevance, namely $\mathcal{A}$ includes all concepts that we are made aware of by the external environment. Then the *maxim of quality* principle gives a **fact scheme**: $\mathsf{fact}(L) = (\emptyset \rhd L) \in \mathcal{A}s$, applied for any statement $L \in \mathcal{F}$ of the current cognitive state $\mathcal{S} = (\mathcal{F}, \mathcal{A})$. Similarly, the *maxim of relevance* principle gives a **hypothesis scheme**: $\mathsf{hyp}(A) = (\emptyset \rhd A) \in \mathcal{A}s$ and $\mathsf{hyp}(\overline{A}) = (\emptyset \rhd \overline{A}) \in \mathcal{A}s$, applied for any proposition, $A \in \mathcal{A}$ of the current cognitive state $\mathcal{S} = (\mathcal{F}, \mathcal{A})$. The two different types of a condition P in relation to a consequent Q, each give a **conditional** argument schemes: When P **is sufficient**: $\mathsf{suff}(P \rightsquigarrow Q) = (P \rhd Q)$ and when P **is necessary**: $\mathsf{necc}(\overline{P} \rightsquigarrow \overline{Q}) = (\overline{P} \rhd \overline{Q})$. Finally, the conflict relation $\mathcal{C}$ is simply that of negation, and the strength relation $\succ$ among the argument schemes is that given above in the cognitive principles.

We will then see that human reasoning in the suppression task can be understood through the dialectic process of argumentation to build acceptable (or admissible) arguments supporting the statement of the question and its

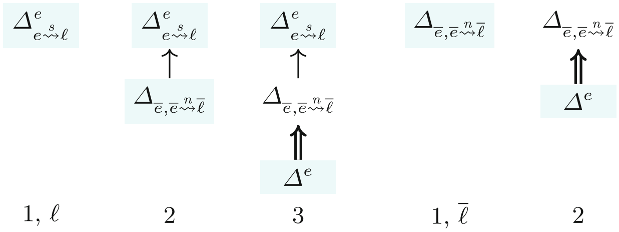

$\Delta^{e}_{e \overset{s}{\leadsto} \ell}$
$\Delta^{e}_{e \overset{s}{\leadsto} \ell}$
$\Delta_{\overline{e}, \overline{e} \overset{n}{\leadsto} \overline{\ell}}$
$\Delta^{e}_{e \overset{s}{\leadsto} \ell}$
$\Delta_{\overline{e}, \overline{e} \overset{n}{\leadsto} \overline{\ell}}$
Δ^{e}
$\Delta_{\overline{e}, \overline{e} \overset{n}{\leadsto} \overline{\ell}}$
$\Delta_{\overline{e}, \overline{e} \overset{n}{\leadsto} \overline{\ell}}$
Δ^{e}
1, ℓ
2
3
1, $\overline{\ell}$
2

Fig. 2. Argumentation process for ℓ and $\overline{\ell}$ in Group I. Only ℓ is acceptable.

negation. Figures 2 and 3 show this for Group I and Group III in terms of the following **dialectic argumentation** process:

Step 1 construct a root argument supporting a conclusion of interest,
Step 2 consider a counterargument against the root argument,
Step 3 find a defense argument against the counterargument,
Step 4 check if this defense argument is not in conflict with the root argument,
Step 5 add this defense argument to the root argument,
Repeat from **Step 2**, with the extended root argument.

Carrying out the process until there are no other counterarguments in **Step 2** that have not already being considered, clearly results in an extended root argument that is an acceptable argument supporting the conclusion of interest.

Figure 2 shows this process to build an argument for ℓ (for Group I) starting with the relatively strong argument of $\Delta^{e}_{e \overset{s}{\leadsto} \ell} = \{\mathsf{fact}(e), \mathsf{suff}(e \leadsto \ell)\}$ (Fig. 2.1, ℓ). This is attacked by the argument $\Delta_{\overline{e}, \overline{e} \overset{n}{\leadsto} \overline{\ell}} = \{\mathsf{hyp}(\overline{e}), \mathsf{necc}(\overline{e} \leadsto \overline{\ell})\}$ supporting $\overline{\ell}$ (Fig. 2.2) but this immediately defended against (or defeated) by $\Delta^{e} = \{\mathsf{fact}(e)\}$ (Fig. 2.3) which attacks $\Delta_{\overline{e}, \overline{e} \overset{n}{\leadsto} \overline{\ell}}$ on the hypothesis part it contains. This strong attack by Δ^{e} which cannot be defended against is the reason why we cannot build an acceptable argument supporting $\overline{\ell}$, as we see in the right part of Fig. 2. Hence, on the one hand $\Delta^{e}_{e \overset{s}{\leadsto} \ell}$ acceptably supports ℓ while there is no acceptable support for $\overline{\ell}$. Consequently, ℓ is a definite conclusion. This conforms with the empirical observation of an overwhelming majority of responses for *She will study late in the library* in this first group (96%).

In contrast, for Group III, Fig. 3 shows how we can build acceptable arguments for either ℓ (left part of the figure) or $\overline{\ell}$ (right part of the figure) using the new argument $\Delta_{\overline{o}, \overline{o} \overset{n}{\leadsto} \overline{\ell}} = \{\mathsf{hyp}(\overline{o}), \mathsf{necc}(\overline{o} \leadsto \overline{\ell})\}$ that is enabled by the awareness, in Group III, of the concept of open and conditional schemes involving this. Hence in Group III both ℓ and $\overline{\ell}$ are acceptably supported and hence are only plausible (credulous) conclusions. This then accounts for the observed suppression effect, where only 38% responded that definitely *She will study late in the library*. Those participants who considered the possibility of the library being not open could support that she did not study in the library and so did not answer that ℓ definitely holds. All twelve cases of the suppression task, where empirical data is collected, can similarly be accounted for in Cognitive Argumentation.

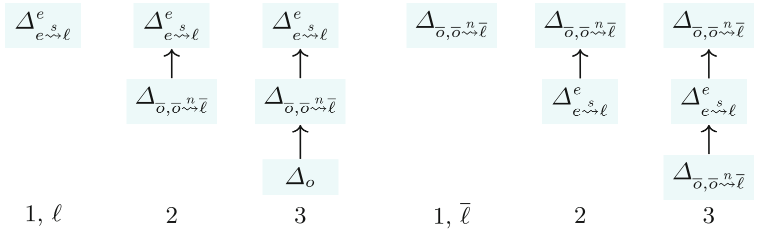

$\Delta^{e}_{e \overset{s}{\rightsquigarrow} \ell}$
$\Delta_{\overline{o},\overline{o} \overset{n}{\rightsquigarrow} \overline{\ell}}$
Δ_{o}
1, ℓ
2
3
1, $\overline{\ell}$
2
3

Fig. 3. Argumentation process for ℓ and $\overline{\ell}$ in Group III. Both ℓ and $\overline{\ell}$ are acceptable.

3.2 The COGNICA System

COGNICA[10] is a system, built on top of the GORGIAS system, that implements the framework of Cognitive Argumentation with emphasis on conditional reasoning. It is based on the particular work of Johnson-Laird and Byrne, "Conditionals: A Theory of Meaning, Pragmatics, and Inference" and the mental models theory that underlies this work. It has a simple interface of a Controlled Natural Language for expressing different types of conditional sentences which are automatically translated into the GORGIAS argumentation framework by adapting and extending the mental models interpretation from a theory on individual conditionals to sets of conditionals and their interaction.

The controlled natural language of COGNICA allows one to enter conditionals in these different types as *foreground knowledge*, i.e., particular knowledge that the system would reason about. Any relevant *background knowledge* is entered in the system, alongside the foreground knowledge, using exactly the same conditional form of controlled natural language.

Example 5 (Foreground Knowledge). Consider the ethics example of "Hal vs Carla" introduced in the tutorial on Argumentation and AI in this school.[11] Its specific foreground knowledge can be captured as:
If use someone's resource **then** compensate.
If justified use of someone's resource **then** not compensate.
If in life threatening situation **then** justified use of someone's resource.
If have alternatives **then** not justified use of someone's resource.

Then given a certain case where the following facts hold, "use of someone's resource", "in life threatening situation" and "have alternatives", the COGNICA system will reply "Maybe" to the query of whether "compensate" holds or not.

COGNICA provides explanations in verbal and graphical form for its answers. Figure 4 shows the graphical explanation for the above answer "Maybe". These

[10] http://cognica.cs.ucy.ac.cy/COGNICAb/login.php.

[11] "Hal, a diabetic, loses his insulin in an accident through no fault of his own. Before collapsing into a coma he rushes to the house of Carla, another diabetic. She is not at home, but Hal enters her house and uses some of her insulin. Was Hal justified, and does Carla have a right to compensation?".

graphical explanations present the argumentative dialectic nature of reasoning by COGNICA as *"reasoning pathways"* of the "mind" of the COGNICA system.

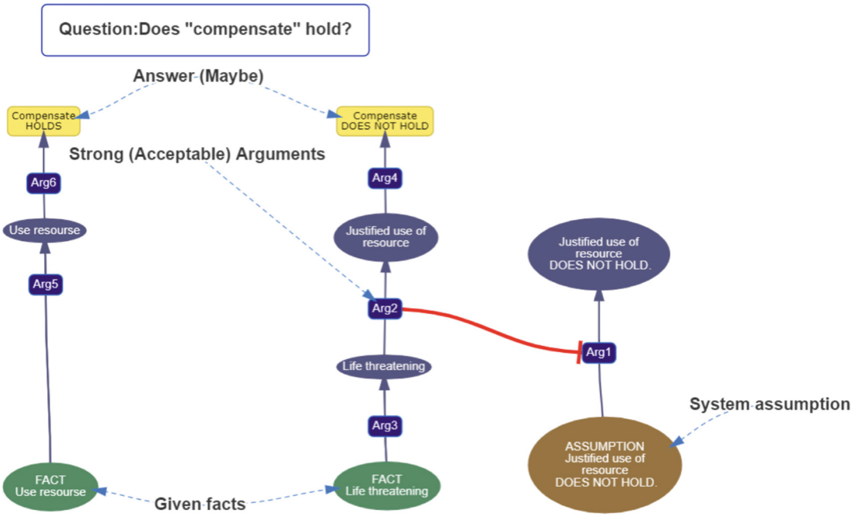

Question:Does "compensate" hold?
Answer (Maybe)
Compensate HOLDS
Compensate DOES NOT HOLD
Strong (Acceptable) Arguments
Arg6
Arg4
Use resourse
Justified use of resource
Justified use of resource DOES NOT HOLD.
Arg5
Arg2
Arg1
Life threatening
System assumption
Arg3
ASSUMPTION Justified use of resource DOES NOT HOLD.
FACT Use resourse
Given facts
FACT Life threatening

Fig. 4. Visual explanation of COGNICA for the "Hal vs Carla" example.

A first evaluation experiment has been set up to evaluate both the "naturality" of the system's conclusions and the possible effect of the system's explanations on the human reasoning. The main part of the experiment consists of each participant shown a short piece of text about a common everyday situation and asked to answer questions on whether a statement holds. The human participant is then shown the answer of COGNICA with its explanations and asked to reconsider her/his answer after seeing these. The initial results of this experiment have shown 70% agreement between human participants and COGNICA on the answers which increases to 85% agreement after seeing the explanation of COGNICA. The change of human's answers occurred mainly when the COGNICA answer was "maybe", and there is a "drift" to more "careful or intense reasoning" by the human participants as they continue. The exercise is open to anyone and can be found at http://cognica.cs.ucy.ac.cy/cognica_evaluation/index.html.

3.3 Argumentation and Cognitive Architectures

The cognitive architecture ACT-R is a theory about how human cognition works. Cognitive functions are represented by modules that communicate with others through buffers. Simulations of these modules and their interactions aim at better understanding processes in human cognition. One strength of ACT-R is that is allows the representation of knowledge symbolically while including sub-symbolic components. Here, we will sketch how cognitive argumentation can be guided by some functionalities in ACT-R. In particular, we discuss the declarative memory, the procedural module, and spreading activation.

Declarative Memory. Declarative memory stores knowledge as chunks, each of them having a name (used for reference) and possibly containing a set of named slots with single values. Consider the following two examples:

```
(ESSAY-SUF isa meaning word "essay" context SUFFICIENT)
(ARGUMENT-FOR-L isa argument fact "essay" position "library"
                    opposite-pos "not library" context SUFFICIENT)
```

The chunk named `ESSAY-SUF` is of type `meaning` and has two slots: `word` has the (string) value `"essay"`, whereas `context` has the value `SUFFICIENT`, which is yet another chunk. The chunk `ARGUMENT-FOR-L` is of type `argument` and has four slots: `fact,` `position`, and `opposite-pos` have the (string) value `"essay"`, `"library"` and `"not library"`, respectively, whereas the slot `context` has as value the chunk `SUFFICIENT`.

```
(p retrieve-word-semantics
    =imaginal>
      word      =word
==>
    +retrieval>
      isa meaning
      word      =word)
```

```
(p retrieve-counter-argument
    =goal>
      state        retrieve-counter
    =retrieval>
      fact         =fact
      position     =position
      opposite-pos =opposite-pos
==>
    +retrieval>
      fact         =fact
      position     =opposite-pos
    =goal>
      state        choose-strongest)
```

Fig. 5. Two simple examples of production rules in ACT-R.

Procedural Module. The procedural module synchronizes the different functionalities in ACT-R and modifies the model's state through the execution of rules. Consider the production rule `retrieve-word-semantics` in Fig. 5 (left): This production rule is only considered if the left hand side (everything before the `==>` sign) is true: there needs to be a slot called `word` in the imaginal buffer with a certain value represented as the variable `=word`. Note that the imaginal buffer can be understood as a place where context information is represented internally. If this rule fires, then the right hand side applies (everything after the `==>` sign): the cognitive model requests a chunk from the `retrieval` buffer, which needs to be of type `meaning` with the slot `word` and has the value `=word`, as defined in the imaginal buffer. Assume that the cognitive model reads the string “essay” which then will be represented internally in its imaginal buffer. If this rule is fired and ESSAY-SUF is in the declarative memory, then ESSAY-SUF matches the request and might be retrieved.

The production rule retrieve-counter-argument in Fig. 5 (right) only applies if the state of the goal buffer is retrieve-counter and the retrieval buffer on the left hand side (everything before ==>) contains a chunk with slots fact, position and opposite-pos. If this rule fires, a new retrieval request will be made, i.e., a chunk is requested to the declarative memory constraint by the following properties: The new retrieval needs to have (1) the same value in the slot fact as the current chunk in the retrieval buffer, and (2) the same value in the slot position as the current chunk in the retrieval buffer has in its opposite-pos slot.

Argument Retrieval Guided by Chunk Activation. Recall the dialectic argumentation process (Steps 1–5 on page 9) described in the previous section: This procedure is computationally intensive because in all the main steps, **Steps 1–3**, a choice is required and all counter arguments need to be considered. Yet, exhaustively searching for arguments does not seem to be cognitively plausible. It is more likely that humans consider only a few arguments, possibly only the most ubiquitous ones. Yet, how to determine these arguments? One possible assumption is that this choice is guided by the context, which in ACT-R can be modeled through the activation of chunks: The activation of a chunk in ACT-R is a numerical value based on the recency and frequency this chunk was previously used, a noise parameter and the spreading activation, i.e., in how far the chunk is related to other chunks in the current context.[12] The chunk’s activation determines whether that chunk will be chosen upon retrieval.

In the current ACT-R implementation, the main arguments are represented as whole chunks. The retrieval of arguments depends on their activation, which is determined by whether the given contexts will rather activate the NECESSARY or SUFFICIENT chunks. Consider the production rule retrieve-counter-argument on page 13: The counter argument with the highest activation will be chosen, and this activation in turn, is determined by the parameters above. For instance, if the chunk SUFFICIENT has a higher activation than the chunk NECESSARY, arguments with the value SUFFICIENT in their context slot (see argument ARGUMENT-FOR-L on page 13) are more likely to be retrieved than arguments with the same slot values for fact and position but where context has the chunk value NECESSARY.

4 Argumentation for Learning

We now continue to discuss the fundamental role of argumentation in the backdrop of the emergent need for Explainable ML, and how argumentation supports

[12] For more information on the activation function see e.g., http://act-r.psy.cmu.edu/wordpress/wp-content/themes/ACT-R/tutorials/unit5.htm.

this role by: *(i)* acknowledging the need to deal with data that is uncertain, incomplete, and inconsistent (with any classical logical theory); *(ii)* offering a target language (syntax and semantics) for learned knowledge that is compatible with human cognition; and *(iii)* supporting a flexible prediction and coverage mechanism for learning that can feed back and guide the learning process.

4.1 What Should a Language of Learning Be Like?

Modern machine learning is typically viewed as a process of turning data into a model that can accurately predict the labels of future data. Increasingly, this focus on predictive accuracy is deemed insufficient as a metric of success, and the ability to explain the reasons behind these predictions is also emphasized.

What counts as an acceptable explanation ultimately boils down to what the purpose of learning is. Learning does not exist, nor carried out, in vacuum, but always takes place in the context of facilitating the informed decision-making of some agent. Learning is coupled with the eventual use of the learned model by the agent, by having each of the two processes guiding and restricting the other. Thus, for example, in situations where a learned model will be used to guide the taking of actions, the coupling implies that learning cannot be done passively.

Learning a model is, thus, not an end but a means to its eventual use. Explanations act as proxy translations of the model into a cognitively-compatible form for the decision-making agent to: *(i)* understand, and adopt or contest, the model's predictions; *(ii)* use predictions and prior knowledge to reach a conclusion; or *(iii)* assimilate the model with prior knowledge in a coherent way.

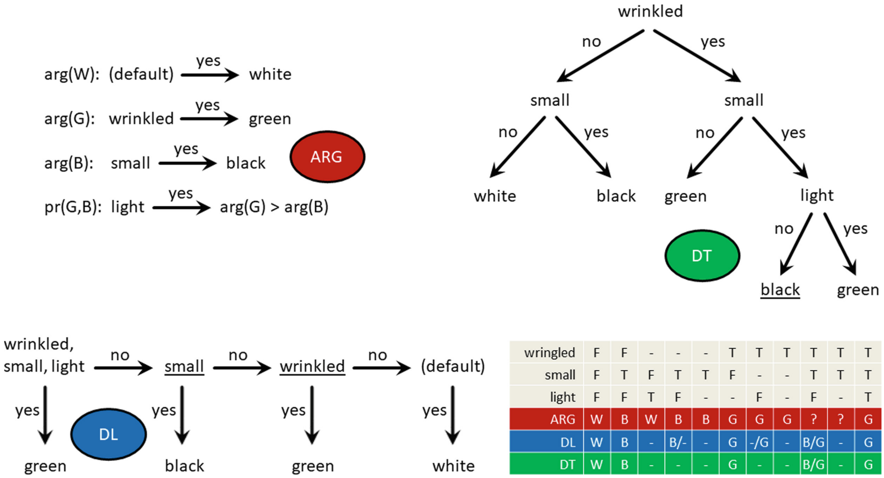
The importance of explanations as proxy translations becomes more apparent in cases of a dilemma: *(i)* on competing predictions of the learned model; *(ii)* on whether we can trust the prediction from a black box; or *(iii)* on how to best utilize or go forward from the prediction. The learned model by itself can not help the decision-making agent to resolve such types of dilemmas, and explanations, then, in support or against the various choices at hand, can help to do so.

The desired characteristics for a language of explanations are none others than those needed to support the role of learning as a facilitator of decision-making: flexibility, adaptability, and ability to recognize and accommodate the inadequacy of the learned model (and the learning process and data) to capture fully the phenomena that produce the data; ability to place the learned model in the cognitive sphere of the decision-making agent; and ability of linking back to the learning process to guide it towards improving the learned model's adequacy.

4.2 Argumentation as a Language of Learning

Below we demonstrate how argumentation can undertake the role for a language of learning and explanations through Pierce's "Beans from a Bag" scenario.

We draw beans from a given bag. Observing that all the drawn beans so far are white, we learn the induced argument arg(W): "beans in this bag are white". If, however, we happen to draw a black bean b1 from the bag, our learned model



wringled	F	F	-	-	-	T	T	T	T	T	T
small	F	T	F	T	T	F	-	-	T	T	T
light	F	F	T	F	-	-	F	-	F	-	T
ARG	W	B	W	B	B	G	G	G	?	?	G
DL	W	B	-	B/-	-	G	-/G	-	B/G	-	G
DT	W	B	-	-	-	G	-	-	B/G	-	G

Fig. 6. Representations of a model learned following the "Beans from a Bag" scenario based on argumentation (ARG), decision trees (DT), or decision lists (DL). Underlined nodes are choices made during learning despite lack of evidence in the training data. The table shows the predictions of the three models on selected data points, where '?' is a dilemma, '-' is an abstention due to missing information in a data point, and pairs of predictions show a dependence on the choices of underlined nodes in learned models.

does not collapse, but is gracefully extended with the observational argument arg(b1): "this particular bean is black". By its nature, an observational argument is stronger than an induced one, naturally accommodating the specific exception or anomaly, while maintaining that all other beans in the bag are white.

As we continue drawing beans from the bag, we might encounter additional non-white beans and learn the induced arguments arg(B): "small beans in this bag are black" and arg(G): "wrinkled beans in this bag are green". Having more specific conditions than arg(W), these two induced arguments are stronger than the latter. So, if we draw again a small bean, then arg(B) will defeat arg(W), and will explain its predictions by *attributing* it to the size of the bean.

The two induced arguments are incomparable, and produce a dilemma in cases of a small wrinkled bean, suggesting that learning needs more such beans to resolve the ambiguity. By drawing additional beans, we might end up learning the priority argument arg(G) > arg(B) if light: "if light bean then green", which does not make a prediction per se, but resolves the dilemma by offering a *contrastive* explanation of why a small wrinkled bean should be green rather than black.

One could posit that other typical white-box representations with some form of prioritization could equally-well take on the role of a language of learning or explanations. Figure 6 shows possible learned models for the scenario above, using argumentation, decision trees, and decision lists, which we compare next.

First, in terms of the representation structure, the conflict resolution process in argumentation is learnable and expressible in a layered fashion. This yields a more compact representation, and avoids imposing a total order or mutual exclusion between conditions. Argumentation does not necessitate access to full information, or even negative information in some cases, and is not over-committed to always reach a prediction if not supported by the statistical evidence from the data. Argumentation can still abstain if the information in any given data point is insufficient, and it will cleanly distinguish an abstention from a dilemma.

Second, in terms of cognitive compatibility, argumentation does not confound the attributive (object-level) explanations from the contrastive (meta-level) ones that defeat conflicting decisions. Argumentation also supports actionable explanations through the elaboration-tolerant amendment of the learned model.

Third, in terms of learning flexibility, argumentation supports the integration of other models/new knowledge, its lack of insistence to firmly predict if not supported by statistical evidence allows it to identify learning gaps for further training, and its natural handling of missing information allows it to encode knowledge and engage in conflict resolution from visible data only.

Despite the natural connection between argumentation and learning, and the diverse ways in which past learning work has used argumentation, this connection remains largely under-explored. This is particularly so in the context of neural-symbolic systems, where conflicts between signals from multiple neural modules could be resolved by an argumentation theory, offering a cognitively-compatible decision-support layer on top of the opaque perception layer, which could help guide the latter's training in a modular and compositional fashion.

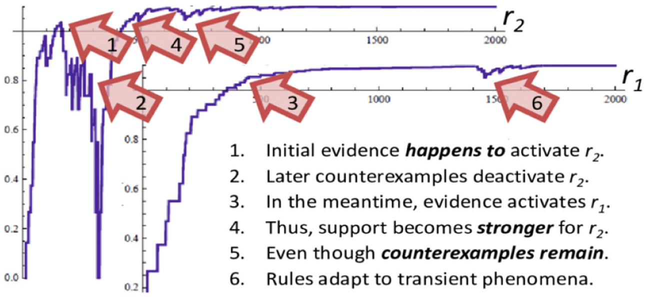
To further appreciate how argumentation and learning can fruitfully interact, we will present two cases of learning with ex ante explainability in mind, where arguments are used natively to represent the learned model and/or data.

4.3 Case Study 1: Autodidactic Learning of Arguments

The first case study that we consider is that of autodidactic (or self-supervised) learning of arguments from partial data, treated as an appearance of some underlying reality, whose commonsense regularities one wishes to learn. These appearances, or observations, are represented as sets of literals; cf. Fig. 8.

The learning mechanism that we consider is called NERD, standing for Never-Ending Rule Discovery. NERD operates in an online/streaming fashion, and passively processes received observations, seeking to identify associations between literals. Confidence in learned rules increases or decreases every time they are satisfied or falsified by an observation. Rules start by being provisional, and become active when their associated confidence exceeds a prescribed threshold.

To resolve conflicts between rules, NERD prioritizes rules based on the order in which they became active, the intuition being that a rule with fewer exceptions (e.g., that penguins cannot fly) will have stronger statistical support from the data, and will become active earlier than a rule with more exceptions (e.g., that birds can fly). Accordingly, when the former rule becomes active, it explains away some of the counter-examples of the latter rule (e.g., observations where birds are also penguins do not count as negative evidence for the latter rule), supporting the latter rule further to gain confidence; see Fig. 7.



1. Initial evidence ***happens to*** activate r_2.
2. Later counterexamples deactivate r_2.
3. In the meantime, evidence activates r_1.
4. Thus, support becomes ***stronger*** for r_2.
5. Even though ***counterexamples remain***.
6. Rules adapt to transient phenomena.

Fig. 7. Updating of the confidence of two learned rules as observations are processed by NERD. The horizontal line in each graph indicates the threshold above which rules are considered active. The arrows show key points of the learning process.

O_{01} = {bird, flying}
O_{02} = {bird, flying}
O_{03} = {penguin, -flying}
O_{04} = {bird, flying}
O_{05} = {bird, penguin}
O_{06} = {bird, penguin, -flying}
O_{07} = {penguin, -flying}
O_{08} = {bird, penguin, -flying}
O_{09} = {bird, penguin}
O_{10} = {penguin}
O_{11} = {penguin, flying}

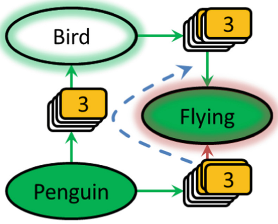


Fig. 8. Observations (left) are iteratively processed by the NERD algorithm to produce the learned model (right). During the last iteration, "penguin" and "flying" are observed (green filled ovals), "bird" and "-flying" are inferred (green and red glowing ovals) by applying the corresponding active rules, and the confidence of the rules "penguin implies not flying" and "bird implies flying" is, respectively, demoted and promoted. The latter rule becomes active (having previously been deactivated from an earlier active state), and is given lower priority than the currently active former rule. (Color figure online)

The interaction between rules happens naturally by simply reasoning with active rules — chaining them together to form arguments, whose strengths come from rule priorities — before each observation is utilized for learning. This approach fully aligns with the coupling of learning with the eventual use of knowledge learned from partial data, as this knowledge is to be used to comprehend observations by completing their missing parts. As NERD proceeds, the learned model increases its coverage with additional (active) rules; see Fig. 8.

4.4 Case Study 2: eXplanations In, eXplanations Out

The second case study that we consider is that of learning by engaging with a user who offers advice to the learner, and from which one wishes to learn a user-specific policy. A learning algorithm processes the feedback coming from a user

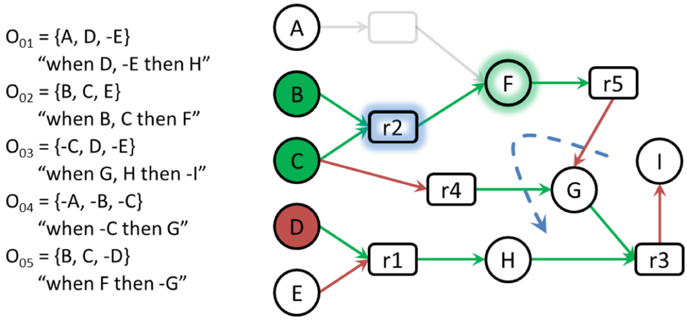


Fig. 9. Contexts and corresponding user reactions (left) are iteratively processed by the Machine Coaching protocol to produce the learned model (right). During the last iteration, "B", "C", and "not D" are observed (green and red filled circles), and "F" is inferred (green glowing circle) by applying the corresponding rule. Following the user's reaction, a new rule $r5$ is added, and is given higher priority than the existing rule $r4$. (Color figure online)

following the eXplanations In, eXplanations Out (XIXO) principle: if we expect to learn a model able to offer explanations that are cognitively compatible with, and acceptable to, a given user, then the same type of explanations should be offered during the learning phase as training material to the learner.

The learning mechanism that we consider is called Machine Coaching, emphasizing the active interaction of the learner with a coach. Machine Coaching operates in an online/streaming fashion, and passively processes received observations. Unlike in the first case study, these observations are not meant to correspond to experiences from which one learns, but rather statements that provide the context within which learning takes place. Given such a context, Machine Coaching proceeds to reason with its existing learned model — following the approach of chaining rules to form arguments from the first case study, and aligning with the coupling of learning — to draw an inference, which it presents to the user along with the arguments in support of that inference.

The user reacts to the inference and the associated explanation of the learned model by offering a counter-argument explaining why the learned model's inference or explanation is not acceptable. Machine Coaching revises the learned model by integrating the user's explanation. This integration happens naturally by virtue of the learned model being represented in the language of argumentation, so that the simple addition of the counter-argument with higher strength than existing conflicting arguments suffices; see Fig. 9. This approach fully aligns with the XIXO principle and the coupling of learning.

Unlike typical online learning, in Machine Coaching the supervision signal is not the label of a data point, nor a reaction to whether the prediction of the current learned model is correct, but rather a reaction to whether the explanation of the learned model is acceptable to the user. On the other hand, the goal of the learned model is not to anticipate what supervision signal it would have gotten on a future data point, but rather to make a prediction and an associated explanation that would lead to no reaction from the user. Finally, note that each supervision signal offers information beyond the individual data point, as it proactively provides information on the labels of multiple future data points (those that satisfy the conditions of the counter-argument), making the process more efficient than a typical supervised learning process. Despite ultimately being a form of machine learning, Machine Coaching can be best understood as lying between learning and programming, with the dialectical exchange of explanations between the learner and the user leading to a better balance between the user's cognitive effort and the learner's computational burden, compared to what one would get at either of the extreme cases of learning and programming.

5 Applications of Argumentation

In this final section we will see how to develop real-life, large scale applications of argumentation based on the theory and methods presented in the earlier parts of the tutorial. We will examine a general methodology for developing argumentation-based systems within a simple high-level architecture and illustrate this with several example applications from various domains. These AI systems are designed with an emphasis on their **(soft) embodiment** within an external dynamic environment with a two-way continual interaction with the environment that includes the "human in the loop". To realize such human-centric AI systems we can follow a human, "mind-like" architecture, as in Fig. 10.

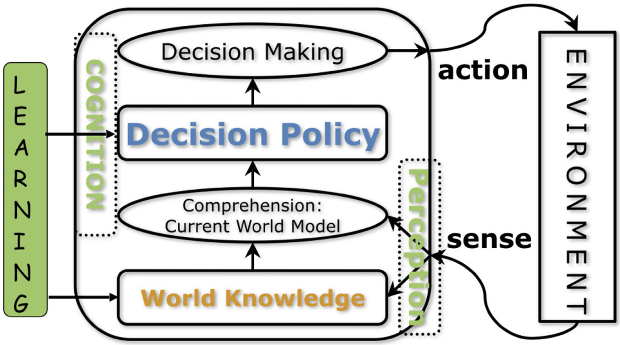


Fig. 10. High-level architecture for Cognitive AI Systems.

In this architecture there are two levels of knowledge that the system utilizes. At the top we have the decision policy containing the specific **application knowledge** of the requirements which regulate the decision making process. This is expressed in terms of high-level concepts about the current state of affairs under which a decision has to be taken and depends on the high-level understanding of the current external environment based on the sensory information that the system has (just) received. Recognizing the information received through the sensors and comprehending this in terms of the higher-level application language is based on **world knowledge**. This knowledge associates the lower-level information from the environment to increasingly higher-levels of generalization or abstraction required by the top application knowledge.

The central task of developing these system rests in acquiring or learning these two pieces of knowledge. This poses two **major challenges**:

- **Acquisition of Application Knowledge** What is an appropriate language level that would facilitate capturing the application knowledge either from the application expert and/or the application data? What is the appropriate cognitive-level of this language?
- **Middleware from Sensory Information to High-level Application Concepts** What are effective ways of comprehending the relevant part of the current application environment? How do we recognize the current state of affairs and the particular decision context in which the system finds itself?

5.1 SoDA: Software Development Through Argumentation

Motivated by the above challenges we can adopt a knowledge representation approach where knowledge is captured in terms of a structure called **Scenario-Based Preference (SBP)**. This is a high-level structure that allows us to represent knowledge directly at the application level in terms of the application language. It can be translated automatically into an executable *Gorgias* argumentation theory thus implementing the decision policy and comprehension modules of our Cognitive AI system.

Scenario-based preferences are triplets, $\langle Id, Scenario, POptions \rangle$ where *Id* is a unique identifier of the triplet, *Scenario* is a set of conditions that partially describe a possible subset of states or scenarios of the application environment and *POptions* is a subset of decision options that are preferred in any state where the *Scenario* conditions hold. As we will see below it is very useful to group these scenarios in hierarchies of increasing specificity. Essentially, scenario-based preferences are a formal structure that allows us to capture knowledge of the general cognitive form:

> "Generally, when [SITUATION] prefer O_i, but in the more particular [CONTEXT], prefer O_j"

where O_i and O_j are subsets of options and SITUATION, CONTEXT are subsets of scenario conditions with CONTEXT describing a more specific situation.

Let us illustrate, through a simple example, SBPs and a methodology, called *SoDA: Software Development through Argumentation*, for acquiring and building the application knowledge as a set of hierarchies of SBPs.

Example 6 (Study Assistant). Consider the problem of deciding where to study with three possible options, study at the *Library*, *Home* or *Cafe*. Assume that we are given or learned the decision guidelines:

"When [Have Homework] prefer to study at *Home*, *Cafe*, but if [Late], prefer to study at *Home* or when [Need Sources] prefer to study at *Library*."

This is captured by following hierarchy of scenario-based preferences:

$$\langle 1, \{\mathit{Homework}\}, \{\mathit{Home}, \mathit{Cafe}\}\rangle \quad \begin{array}{l} \langle 11, \{\mathit{Homework}, \mathit{Late}\}, \{\mathit{Home}\}\rangle \\ \langle 12, \{\mathit{Homework}, \mathit{Need\_Sources}\}, \{\mathit{Library}\}\rangle \end{array}$$

Here each of 11 and 12 form **refinements** of the root scenario-based preference 1 resulting into two the hierarchies of (1,11) and (1,12).

Together with the operation of refinement, we have a second operation of **combination**, where we consider the union of scenario conditions from two SBPs. From Example 6 consider the combination of scenarios in 11 and 12 to generate the interim new SBP of: $\langle 11|12i, \mathit{Homework}, \mathit{Late}, \mathit{Need\_Sources}\}$, $\{\mathit{Home}, \mathit{Library}\}\rangle$.

We can then return to the decision policy, e.g., ask or learn from the application owner or user, for possible preferences in the combined scenario and generate SBPs refining the interim SBP. For example, we may learn a preference to *Library* and so have: $\langle 11|12, \{\mathit{Homework}, \mathit{Late}, \mathit{Need\_Sources}\}, \{\mathit{Library}\}\rangle$.

The *SoDA* methodology provides guidelines for carrying out this process of knowledge engineering of scenario-based preferences. An associated tool, called **Gorgias-B**[13], supports the methodology by providing a framework to build contextual refinements of SBPs and to consider appropriate combinations of these. This tool also carries out an automatic generation, from the SBPs, of an argumentation theory in the *Gorgias* framework and an interface to execute this under the GORGIAS system. This has now evolved into a professional platform tool, called *rAIson*, developed by a new company, called *Argument Theory*[14].

5.2 Application Language Levels: Example Applications

The language that we use to specify the decision policy of an application can vary according to the nature of the application. Ideally, the level of the application language should be as close as possible to natural language or some form of structured natural language. Given a language level the task for translating a decision policy into scenario-based preferences differs in the degree of manual effort required. The following is a list of different application language levels each together with a typical real-life application domain.

- **Free Text in Structured Natural Language**.
  An example case is that where the policy is given in a **Legal Document**. Such documents are highly structured and are already in a scenario-based preference and argumentation form. The translation into a scenario-based preference form is carried out manually but this is direct and it is easily

[13] http://gorgiasb.tuc.gr/.
[14] https://www.argument-theory.com/en/.

carried out. We can then automate the task of **compliance** with the legal requirements providing explanations of why an action is compliant or not and if not how it can become compliant. An example of such application is MEDICA[15] a system for granting the appropriate level of access to the electronic patient record, as specified by the European law.

- **Controlled Natural Language in a Restricted Vocabulary**
  This language level is appropriate for **Cognitive Assistant** applications where these systems provide services in a restricted domain of interest. Examples of such cognitive assistants are Call Assistant, Tourist Assistant, Care Assistant, Calendar Assistant, Investor Assistant and Social Media Assistant. These systems can start with a minimal vocabulary and gradually expand it as the systems are starting to be deployed. The policy guidelines are given in a controlled form of natural language customized to the vocabulary and particular features of the domain of application.
  Let us consider the example of a Social Media assistant. This is a system that monitors the user's social media feed and helps a user manage the information overload by explainably "re-arranging", according to the user's liking, the information pieces, e.g., posts, that she/he receives. For example, for each post the assistant would decide amongst highlighting this at the top or even notifying the user, demoting it to the bottom or hiding it completely and other such actions. A user can express her/his personal policy guidelines at a high level using controlled natural language. For example:

  > *I like sports, particularly tennis and basketball. I love drama and comedy movies especially if produced in the UK. I like to know what my closest friends are doing and to stay in touch with current popular news. But I hate politics except when related to climate change.*

  Representing this in scenario-based preferences we take into consideration two types of information conveyed in this type of policies: (1) the high-level concepts that act as decision criteria, e.g., "sports", "drama movies", "produced in the UK", "closest friends", "popular news", ..., that form the scenario conditions and (2) the implicit preferences conveyed by various keywords used, e.g., "like", "love", "particularly" "stay in touch", "hate", "except", ..., used to fill in the preferred options in scenario-based preferences and to form refinements of these. The sensory information received by the social media assistant is the low-level data on each post that the user receives on a social media platform, such as who posted it, its content, its popularity figures, etc. We can then build *middleware* based on different technologies to decide on the description of the post in terms of the high-level concepts referred to in the decision policy. The output of the assistant is a presentation of the user's posts based on which classification and related action can be supported acceptably by the underlying *Gorgias* argumentation theory. The classification is shown next to the post together with its supporting explanation when the user wishes to see it. A typical example explanation, that highlights their contrasting nature is:

[15] http://medica.cs.ucy.ac.cy/.

> "Even though this post is not (very) interesting to you it was made from a close friend."

- **Structured Tables of Scenario Hierarchies**
  For applications that are based on expert knowledge, e.g., **Decision Support Assistants**, we need a more structured language to capture large scale amounts of expert knowledge. A suitable such structure is that of *structured tables* where each row essentially corresponds to a scenario-based preference. The first column of the table contains the scenario conditions and each of the other columns corresponds to a single option which is marked or not in each row as one of the preferred options in the scenario of the row.
  In the medical domain, where this has been mostly applied, the doctors use their familiar medical language for constructing/filling these tables. Also they are already familiar, from Evidence Medicine, with the notions of supporting and differential evidence, which are directly used in the construction of these tables. This method for knowledge acquisition has been applied to two particular cases of medical decision support, in ophthalmology and in the much larger domain of gynecology. The purpose of the first system of *OPHTALMOLOGICA*, is to understand the level of severity of the possible disease(s) so that a scheduling appointment system (or the receptionist) can give an appropriate priority to the patient. The second system of *GAID: Gynecological AI Diagnostic Assistant* has its overall aim to:

  > *"Support clinicians feel more confident in decision, helping to avoid over-diagnosis of common diseases and to ensure that emergency cases are not missed out."*

  It covers fully the area of gynecology with over 140 diseases (i.e., options) and over a thousand different parameters (current symptoms, patient record, clinical examination findings and laboratory tests) that can affect the diagnosis. The system generates a set of suspicious diseases every time some new information about a patient is received (during a clinical visit to the doctor). All suspicious diseases come with an explanation, generated automatically from the *Gorgias* object-level and priority arguments that support the suspicion of the disease. A typical example of an explanation is:

  > "Under the information *Vaginal Burning* it is **recommended** that you investigate *Vulva Candidiasis.* This is also **supported** by *Post-Coital Bleeding* and further **strengthened** by *Vaginal Discharge.* A negative test for *Vaginal Secretions* would **exclude** this disease."

  The GAID system is under a pilot clinical trial to evaluate both the accuracy of its suggested suspicious diseases as well as its guidance to (junior) clinicians to collect relevant information that would help focus the diagnosis.

5.3 Machine Learning Assisted Policy Formation

Machine Learning offers the possibility to automatically acquire (at least partly) the knowledge for Cognitive AI systems with the high-level architecture of

Fig. 10. As presented in Sect. 4, treating learned associations as object-level arguments we can continue the learning process to construct priority arguments over these thus improving the predictive power of our learned theory and providing more informed explanations. Importantly, by adopting this argumentation perspective on learning we can then integrate together with the machine learned knowledge other knowledge that is already known by experts (e.g., medical or clinical knowledge) and thus have a hybrid approach in generating and updating the knowledge of our application. Machine learning is thus integrated with knowledge elicitation methods directly from the "policy source/owner" as we saw above. Indeed in many expert application cases, but also in other application domains, we can have domain expert knowledge, or a company's business policy or a legal requirement, that it would be futile to insist to learn again through machine learning on a large data corpus of example cases.

Examples of applications where we have machine learning assisted knowledge generation or acquisition have been developed in the area of medical decision support area based on real-life data sets in the area of risk assessment of Stroke and the area of deciding on the possible development of Alzheimer. These systems act as *peer companions* to the doctors to offer a second opinion on a new case. This is done through **"peer explanations"** at the cognitive level of the specialists, e.g., the radiologist or doctor, offered by the system. An example of such a "peer explanation" for the domain of Stroke is:

> "This patient is judged to be *asymptotic* **because** Log(GSM+40) is in the range [4.28, 5.17] and has no history of contralateral TIAs or Stroke. **Although** the patient has (Plaque Area)1/3 in the range [3.47, 6.78] and Discrete White Areas in the plaque, suggesting a risk for *stroke*, the first symptoms are **more significant when** the patient has (Plaque Area)1/3 less than 3.9."

Finally, we mention the area of **argument mining** which in effect uses machine learning to extract arguments (mostly from text) to construct knowledge in the form of argument graphs. This is particularly appropriate for learning the world knowledge on which we base the middle-ware of an application that links sensory information to high-level cognitive concepts on which the decision policy is formed. Argument mining is not covered in this tutorial but it is very important and the reader is urged to consult the many references on this topic.

References

1. Besnard, P., et al.: Tutorials on structured argumentation. Argument Comput. **5**(1), 1–4 (2014)
2. Kakas, A.C., Moraitis, P.: Argumentation based decision making for autonomous agents. In: Proceedings of 2nd International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS, pp. 883–890. ACM (2003)
3. Prakken, H.: An abstract framework for argumentation with structured arguments. Argument Comput. **1**(2), 93–124 (2010)
4. Anderson, J.R.: How Can the Human Mind Occur in the Physical Universe? Oxford University Press, Oxford (2007)

5. Byrne, R.: Suppressing valid inferences with conditionals. Cognition **31**, 61–83 (1989)
6. Dietz, E., Kakas, A.C.: Cognitive argumentation and the suppression task. CoRR abs/2002.10149 (2020). https://arxiv.org/abs/2002.10149
7. Michael, L.: Autodidactic learning and reasoning. doctoral dissertation. Harvard University, Cambridge (2008)
8. Michael, L.: Cognitive reasoning and learning mechanisms. In: Proceedings 4th BICA International Workshop on Artificial Intelligence and Cognition, pp. 2–23 (2016)
9. Michael, L.: Machine coaching. In: Proceedings 2019 IJCAI Workshop on Explainable Artificial Intelligence, pp. 80–86 (2019)
10. Almpani, S., Kiouvrekis, Y., Stefaneas, P.: Modeling of medical devices classification with computational argumentation. In: 2021 12th International Conference on Information, Intelligence, Systems Applications (IISA), pp. 1–6 (2021)
11. Kakas, A.C., Moraitis, P., Spanoudakis, N.: Gorgias: applying argumentation. Argument Comput. **10**(1), 55–81 (2019)