# Refining Concepts by Machine Learning

Marek MENŠÍK[1], Marie DUŽÍ[1], Adam ALBERT[1], Vojtěch PATSCHKA[1], Miroslav PAJR[2]
[1]*VSB-Technical University Ostrava, Department of Computer Science FEI*
*17. listopadu 15, 708 33 Ostrava, Czech Republic*
[2]*Silesian Univerzity in Opava, Institute of Computer Science,*
*Bezručovo nám. 13, 746 01 Opava, Czech Republic*

**Abstract.** In this paper we deal with machine learning methods and algorithms applied in learning simple concepts by their refining. The method of refining a simple concept of an object *O* consists in discovering a molecular concept that defines the same object *O*. Typically, such a molecular concept is a professional definition of the object, for instance a biological definition according to taxonomy, or legal definition of roles, acts, etc. Our background theory is Transparent Intensional Logic (TIL). In TIL concepts are explicated as abstract procedures encoded by natural language terms. These procedures are defined as six kinds of TIL constructions. First, we briefly introduce the method of learning with a supervisor that is applied in our case. Then we describe the algorithm 'Framework' together with heuristic methods applied by it. The heuristics is based on a plausible supply of positive and negative (near-miss) examples by which learner's hypotheses are refined and adjusted. Given a positive example, the learner refines the hypothesis learnt so far, while a near-miss example triggers specialization. Heuristic methods deal with the way refinement, including its special cases generalization and specialization, is applied.

**Keywords.** Machine learning, supervisor, Transparent Intensional Logic, TIL, refinement, generalization, specialization, hypothesis, heuristics

## 1. Introduction

The method of supervised machine learning enables the agents in a multi-agent system to adjust their ontology and increase their knowledge. In [12] the method has been applied to learning the concept of a property that classifies geometric figures such as lancet arches. In this paper we deal with *natural language processing*, which is an interdisciplinary discipline involving linguistics, logic and computer science. The goal of this paper is to describe the application of machine learning methods in agents' learning simple concepts by their refinement. Our background theory is Transparent Intensional Logic (TIL) with its *procedural* (as opposed to set-theoretical) semantics. In TIL we explicate concepts procedurally. They are abstract structured *procedures* assigned to natural language terms as their meanings. In this way *structured meanings* are formalized in a fine-grained way as so-called TIL *constructions* so that almost all the semantically salient features can be successfully dealt with. To this end we use the so-called Normal Translation Algorithm (NTA) that processes text data and produces TIL

constructions as their meanings.[1] Having a meaning procedure, we can apply logic to prove what is entailed by it, compute the object (if any) produced by the procedure, deal with its structure, etc.

However, there is a problem of understanding simple or atomic concepts that are expressed by semantically simple terms like 'cat', 'dog', 'myopia', etc. They are basic 'building blocks' of molecular concepts, and as such they are formalized just by the simplest procedure Trivialization of a given object $O$, '$^0O$' in symbols, that refers to the object $O$ and makes it available to other molecular procedures to operate on it. In proof-theoretic semantics the meaning of atomic terms is given by the rules that determine how to use them in proofs.[2] This works well in the language of mathematics and logic. However, in natural language the 'meaning as proof' semantics is much less successful. For these reasons we decided to apply supervised machine learning methods. The issue is this. When processing a natural language text, our agents learn structured TIL procedures encoded by sentences. For instance, the sentence "Tom has myopia" translates into the TIL procedure $\lambda w \lambda t\ [{}^0Myopia_{wt}\ {}^0Tom]$. It can be viewed as an instruction how in any possible world ($\lambda w$) and time ($\lambda t$) evaluate the truth-conditions of the sentence, which consists of these steps:

- Take the individual Tom: $^0Tom$
- Take the property of having Myopia: $^0Myopia$
- Extensionalize the property with respect to world $w$ and time $t$ of evaluation: $^0Myopia_{wt}$
- Produce a truth-value by checking whether Tom has this property at the world $w$ and time $t$ of evaluation: $[{}^0Myopia_{wt}\ {}^0Tom]$

So far so good. We can derive that somebody has myopia, but this piece of information does not suffice to derive, for instance, that Tom has problems with impaired vision, needs negative dioptre correction, etc. We need to refine the simple concept $^0Myopia$ to learn in more details what 'myopia' means. In other words, we want to define the property of having myopia. To this end we try to extract from natural language texts the collection of so-called *requisites* that together define the property. Hence, the supervisor looks for sentences like "Myopia (also called near-sightedness) is the most common cause of impaired vision in people under age 40". Based on this piece of information the agent makes a *hypothesis* that among the requisites of myopia there are 'near-sightedness' and 'impaired vision'. This is a positive example. Furthermore, we can read sentences like "Myopia is not caused by nerve trauma; rather, it occurs when the eyeball is too long, relative to the focusing power of the cornea and lens of the eye. This causes light rays to focus at a point in front of the retina, rather than directly on its surface. Near-sightedness also can be caused by the cornea and/or lens being too curved for the length of the eyeball. In some cases, myopia is due to a combination of these factors." The supervisor should extract a negative example that myopia is not caused by nerve trauma and a collection of positive examples like 'too long eyeball', 'wrong focusing', etc.

The algorithm of the learning process is based on such positive and negative examples. Given a positive example, refinement is applied on the hypotheses so that concepts of other requisites or typical properties are inserted. Negative (also 'near-miss')

[1] For details, see [9], [11].

[2] See, for instance, [8].

examples serve to the adjustment of the hypothesis (learnt so far) by specialization that excludes non-plausible elements. As a special case of refinement, we can also apply generalization. This is the case of inserting a more general concept in addition to some special constituents of the hypothesis. For instance, the degree of myopia is described in terms of the power of the ideal correction, which is measured in dioptres. Now the agent can extract information like this. "Low myopia usually describes myopia of -3.00 dioptres or less (i.e. closer to 0.00), moderate myopia is between -3.00 and -6.00 dioptres, and high myopia is the degree -6.00 or more." By generalization we obtain information that myopia is corrected by negative dioptres.

The rest of the paper is organized as follows. In Section 2 we summarize foundations of TIL to describe logical machinery that we need in the rest of the paper. Section 3 introduces the principles of supervised machine learning. In Section 4 we deal with heuristic methods that are used to adjust and enrich agents' knowledge base. In Section 5, an example of using the algorithm of machine learning together with TIL formalization is adduced. Finally, concluding remarks can be found in Section 6.

## 2. Foundations of Transparent Intensional Logic (TIL)

Since the TIL logical system has been introduced in numerous papers and two books, see, for instance [1], [2], [3], [4], [5], [6], [7], [16], here we just briefly summarise the main principles of a TIL fragment that we need for the purposes of this paper.

TIL is a partial, typed hyperintensional lambda calculus with *procedural* as opposed to set-theoretical denotational semantics. The terms of the TIL language denote abstract procedures that produce set-theoretical mappings (functions-in-extension) or lower-order procedures. These procedures are rigorously defined as TIL *constructions.* Being procedural objects, constructions can be executed in order to operate on input objects (of a lower-order type) and produce the object (if any) they are typed to produce, while non-procedural objects, i.e. non-constructions, cannot be executed. There are two atomic constructions that present input objects to be operated on. They are *Trivialization* and *Variables.* The operational sense of Trivialization is similar to that of constants in formal languages. The Trivialization presents an object $X$ without the mediation of any other procedures. Using the terminology of programming languages, the Trivialization of $X$, '$^0X$' in symbols, is just a *pointer* that refers to $X$. Variables produce objects dependently on valuations; they *v*-construct. We adopt an objectual variant of the Tarskian conception of variables. To each type countably many variables are assigned that range over this particular type. Objects of each type can be arranged into infinitely many sequences. The valuation $v$ selects one such sequence of objects of the respective type, and the first variable *v*-constructs the first object of the sequence, the second variable *v*-constructs the second object of the sequence, and so on. Thus, the execution of a Trivialization or a variable never fails to produce an object. However, the execution of some of the molecular constructions can fail to present an object of the type they are typed to produce. When this happens, we say that the constructions are *v-improper*.

There are two dual molecular constructions which correspond to λ-abstraction and application in λ-calculi, namely *Closure* and *Composition*. (λ-)Closure, $[\lambda x_1 \ldots x_n\ X]$, transforms into the very procedure of producing a function by abstracting over the values of the variables $x_1, \ldots, x_n$. The Closure $[\lambda x_1 \ldots x_m\ Y]$ is not *v*-improper for any valuation $v$, as it always *v*-constructs a function. Composition, $[X\ X_1 \ldots X_n]$, is the very procedure of applying a function produced by the procedure $X$ to the tuple-argument (if any) produced

by the procedures $X_1$, …, $X_n$. While Closure never fails to produce a function, Composition is *v*-improper if one or more of its constituents $X, X_1, \ldots, X_n$ are *v*-improper. This happens when a partial function *f* is applied to an argument *a* such that the function *f* is not defined at *a.* Another cause of improperness can be type-theoretical incoherence of the Composition. For instance, the proposition that the number 5 is a student does not have a truth-value at any world *w* and time *t* of evaluation, because the property of being a student is the property of individuals rather than numbers. Hence the application of the (extensionalized) property of being a student to the number 5 in a particular world *w* and time *t* of evaluation, in symbols $[[[^0\textit{Student}\ w]t]\ {}^0 5]$, or $[{}^0\textit{Student}_{wt}\ {}^0 5]$ for short, is *v*-improper for every valuation *v* of the variables *w* (ranging over possible worlds) and *t* (ranging over times).

**Definition (*constructions*)**

(i) *Variables* $x, y, \ldots$ are *constructions* that *construct* objects (elements of their respective ranges) dependently on a valuation *v*; they *v-construct*.

(ii) Where *X* is an object whatsoever (even a *construction*), ${}^0X$ is the *construction Trivialization* that *constructs X* without any change of *X*.

(iii) Let $X, Y_1, \ldots, Y_n$ be arbitrary *constructions*. Then *Composition* $[X\ Y_1 \ldots Y_n]$ is the following *construction*. For any *v*, the *Composition* $[X\ Y_1 \ldots Y_n]$ is *v-improper* if at least one of the *constructions* $X, Y_1, \ldots, Y_n$ is *v*-improper by failing to *v*-construct anything, or if *X* does not *v-construct* a function that is defined at the *n*-tuple of objects *v-constructed* by $Y_1,\ldots,Y_n$. If *X* does *v-construct* such a function, then $[X\ Y_1 \ldots Y_n]$ *v-constructs* the value of this function at the *n*-tuple.

(iv) (λ-) *Closure* $[\lambda x_1 \ldots x_m\ Y]$ is the following *construction*. Let $x_1, x_2, \ldots, x_m$ be pair-wise distinct variables and *Y* a *construction*. Then $[\lambda x_1 \ldots x_m\ Y]$ *v-constructs* the function *f* that takes any members $B_1, \ldots, B_m$ of the respective ranges of the variables $x_1, \ldots, x_m$ into the object (if any) that is $v(B_1/x_1,\ldots,B_m/x_m)$-*constructed* by *Y*, where $v(B_1/x_1,\ldots,B_m/x_m)$ is like *v* except for assigning $B_1$ to $x_1, \ldots, B_m$ to $x_m$.

(v) Where *X* is an object whatsoever, ${}^1X$ is the *construction Single Execution* that *v*-constructs what *X v-constructs*. Thus, if *X* is a *v*-improper *construction* or not a *construction* as all, ${}^1X$ is *v-improper*.

(vi) Where *X* is an object whatsoever, ${}^2X$ is the *construction Double Execution*. If *X* is not itself a *construction*, or if *X* does not *v*-construct a *construction*, or if *X v-constructs* a *v*-improper *construction*, then ${}^2X$ is *v-improper*. Otherwise ${}^2X$ *v-constructs* what is *v-constructed* by the *construction v-constructed* by *X*.

Nothing is a *construction*, unless it so follows from (i) through (vi).

With constructions of constructions, constructions of functions, functions, and functional values in our stratified ontology, we need to keep track of the traffic between multiple logical strata. The *ramified type hierarchy* does just that. The type of first-order objects includes all objects that are not constructions. Therefore, it includes not only the standard objects of individuals, truth-values, sets, etc., but also functions defined on possible worlds (i.e., the intensions germane to possible-world semantics). The type of second-order objects includes constructions of first-order objects and functions that have such construction in their domain or range. The type of third-order objects includes constructions of first- and second-order objects and functions that have such construction in their domain or range. And so on, ad infinitum.

**Definition (*types of order n*).** Let $B$ be a *base*, where a base is a collection of pair-wise disjoint, non-empty sets. Then:

**$T_1$** *(types of order 1)*.

i) Every member of $B$ is an elementary *type of order 1 over B.*

ii) Let $\alpha, \beta_1, ..., \beta_m$ ($m > 0$) be types of order 1 over $B$. Then the collection $(\alpha\ \beta_1 ...\ \beta_m)$ of all $m$-ary partial mappings from $\beta_1 \times ... \times \beta_m$ into $\alpha$ is a functional *type of* order 1 *over B.*

iii) Nothing is a *type of order 1 over B* unless it so follows from (i) and (ii).

**$C_n$** *(constructions of order n)*

i) Let $x$ be a variable ranging over a type of order $n$. Then $x$ is a *construction of order n over B.*

ii) Let $X$ be a member of a type of order $n$. Then $^0X$, $^1X$, $^2X$ are *constructions of order n over B.*

iii) Let $X, X_1, ..., X_m$ ($m > 0$) be constructions of order $n$ over $B$. Then $[X\ X_1 ...\ X_m]$ is a *construction of order n over B.*

iv) Let $x_1, ..., x_m, X$ ($m > 0$) be constructions of order $n$ over $B$. Then $[\lambda x_1 ... x_m\ X]$ is a *construction of order n over B*.

v) Nothing is a *construction of order n over B* unless it so follows from **$C_n$** (i)-(iv).

**$T_{n+1}$** (*types of order* $n + 1$)

Let $*_n$ be the collection of all constructions of order $n$ over $B$. Then

i) $*_n$ and every type of order $n$ are *types of order* $n + 1$.

ii) If $m > 0$ and $\alpha, \beta_1, ..., \beta_m$ are types of order $n + 1$ over $B$, then $(\alpha\ \beta_1 ...\ \beta_m)$ (see $T_1$ ii)) is a *type of order* $n + 1$ *over B*.

iii) Nothing is a *type of order n + 1 over B* unless it so follows from (i) and (ii). □

*Remark.* For the purposes of the analysis of our sample example of agents' learning the concept of myopia intensional fragment of TIL based on the simple types of order 1 suffices. Yet when the agents learn new concepts, they enrich their ontology by new constructions that are just displayed rather than executed. To this end, the full ramified hierarchy is needed. For details see, e.g., [4], [7].

For the purposes of natural-language analysis, we are usually assuming the following base of elementary types:

| | |
|---|---|
| o: | the set of truth-values {**T, F**}; |
| ι: | the set of individuals (the universe of discourse); |
| τ: | the set of real numbers (doubling as discrete times); |
| ω: | the set of logically possible worlds (the logical space). |

We model sets and relations by their characteristic functions. Thus, for instance, $(o\iota)$ is the type of a set of individuals, while $(o\iota\iota)$ is the type of a relation-in-extension between individuals. Empirical expressions denote *empirical conditions* that may or may not be satisfied at the world/time pair selected as points of evaluation. We model these empirical conditions as possible-world-semantic *intensions.* Intensions are entities of type $(\beta\omega)$: mappings from possible worlds to an arbitrary type $\beta$. The type $\beta$ is frequently the type of the *chronology* of $\alpha$-objects, i.e., a mapping of type $(\alpha\tau)$. Thus $\alpha$-intensions are frequently functions of type $((\alpha\tau)\omega)$, abbreviated as '$\alpha_{\tau\omega}$'. *Extensional entities* are entities of a type $\alpha$ where $\alpha \neq (\beta\omega)$ for any type $\beta$.

Hence, e*mpirical expressions* denote (non-trivial, i.e. non-constant) intensions. Where variable $w$ ranges over $\omega$ and $t$ over $\tau$, the following logical form essentially characterizes the logical syntax of empirical language:

$$\lambda w \lambda t\, [\ldots w \ldots . t \ldots]$$

Examples of frequently used intensions are:

- *propositions* of type $o_{\tau\omega}$ denoted by sentences like "John is a student";
- *properties of individuals* of type $(o\iota)_{\tau\omega}$ denoted by nouns and adjectives, e.g. 'student', 'red', 'tall', 'myopia', 'near-sighted';
- *binary relations-in-intension between individuals* of type $(o\iota\iota)_{\tau\omega}$, e.g. being 'composed of', 'seeing';
- *individual offices* (or *roles*) of type $\iota_{\tau\omega}$ that are denoted by definite descriptions like 'the tallest mountain', 'Miss World 2019', 'the President of Zanzibar'.

Logical objects like *truth-functions* and are extensional: $\wedge$ (conjunction), $\vee$ (disjunction) and $\supset$ (implication) are of type $(ooo)$, and $\neg$ (negation) of type $(oo)$.

The *quantifiers* $\forall^\alpha$, $\exists^\alpha$ are type-theoretically polymorphic total functions of type $(o(o\alpha))$, for an arbitrary type $\alpha$, defined as follows. The *universal quantifier* $\forall^\alpha$ is a function that associates a class $A$ of $\alpha$-elements with **T** if $A$ contains all elements of the type $\alpha$, otherwise with **F**. The *existential quantifier* $\exists^\alpha$ is a function that associates a class $A$ of $\alpha$-elements with **T** if $A$ is a non-empty class, otherwise with **F**.

*Notational conventions.* Below all type indications will be provided outside the formulae in order not to clutter the notation. Moreover, the outermost brackets of Closures will be omitted whenever no confusion arises. Furthermore, '$X/\alpha$' means that an object $X$ is (a member) of type $\alpha$. '$X \to_v \alpha$' means that $X$ is typed to $v$-construct an object of type $\alpha$, regardless of whether $X$ in fact constructs anything. We write '$X \to \alpha$' if what is $v$-constructed does not depend on a valuation $v$. Throughout, it holds that the variables $w \to_v \omega$ and $t \to_v \tau$. If $C \to_v \alpha_{\tau\omega}$ then the frequently used Composition $[[C\ w]\ t]$, which is the intensional descent (a.k.a. extensionalization) of the $\alpha$-intension $v$-constructed by $C$, will be encoded as '$C_{wt}$'. When applying quantifiers, we use a simpler notation '$\forall x\ B$', '$\exists x\ B$' instead of the full notation '$[{}^0\forall^\alpha\ \lambda x\ B]$', '$[{}^0\exists^\alpha\ \lambda x\ B]$', $x \to \alpha$, $B \to o$, to make the quantified constructions easier to read. When applying truth-functions we use infix notation without Trivialization. For instance, instead of the Composition '$[{}^0\wedge\ A\ B]$' we write simply '$[A \wedge B]$'.

For illustration, here is an example of the analysis of a simple sentence "John is near-sighted". First, type-theoretical analysis, i.e. assigning types to the objects that receive mention in the sentence: *John*/$\iota$; *Nearsighted*/$(o\iota)_{\tau\omega}$; the whole sentence denotes a proposition of type $o_{\tau\omega}$. Now we compose constructions of these objects to construct the denoted proposition. To predicate the property of being near-sighted of John, the property must be extensionalized first: $[[{}^0Nearsighted\ w]\ t]$, or ${}^0Nearsighted_{wt} \to_v (o\iota)$, for short. The Composition $[{}^0Nearsighted_{wt}\ {}^0John] \to_v o$; and finally, the whole empirical sentence denotes a proposition of type $o_{\tau\omega}$, hence it encodes as its meaning the Closure

$$\lambda w \lambda t\, [{}^0Nearsighted_{wt}\ {}^0John] \to o_{\tau\omega}.$$

In TIL we reject individual essentialism; instead, we adhere to *intensional essentialism*. It means that each $\alpha$-intension $P$ is necessarily related to a collection of

*requisites* of *P*, its *essence*, that together define the intension *P.* For instance, requisites of the property of being a horse are the property of being a mammal of the family Equidae, species Equus Caballus, the property of having blood circuit, being a living creature, and many others. Necessarily, if some individual *a* happens to be a horse then *a* is a mammal of the family Equidae, etc.

The requisite relations *Req* are a family of relations-in-extension between two intensions, hence of the polymorphous type $(o\alpha_{\tau\omega}\beta_{\tau\omega})$, where possibly $\alpha = \beta$. Infinitely many combinations of *Req* are possible, but the following is the relevant one that we need for our purpose:[3]

*Req* /$(o(o\iota)_{\tau\omega}(o\iota)_{\tau\omega})$: an individual property is a requisite of another such property.

Thus, we define:

**Definition (*requisite relation between ι-properties*)** Let *X*, *Y* be constructions of properties, *X*, *Y*/$*_n \rightarrow (o\iota)_{\tau\omega}$; $x \rightarrow \iota$, *True*/$(oo_{\tau\omega})_{\tau\omega}$: the property of propositions of being true in a given world and time of evaluation. Then

$$[^0Req\ Y\ X] = \forall w \forall t\, [\forall x\, [[^0True_{wt}\, \lambda w \lambda t\, [X_{wt}\, x]] \supset [^0True_{wt}\, \lambda w \lambda t\, [Y_{wt}\, x]]]].\ \square$$

Gloss *definiendum* as, "*Y* is a requisite of *X*", and *definiens* as, "Necessarily, at every ⟨*w*, *t*⟩, whatever *x* instantiates *X* at ⟨*w*, *t*⟩ also instantiates *Y* at ⟨*w*, *t*⟩."

*Remark*. Here we have to apply the property of propositions *True* to handle partiality. This is due to the fact that there is a stronger relation between properties, namely that of *pre-requisite.* If *Y* is a pre-requisite of *X*, then if an individual *x* does not instantiate *Y* it is neither true nor false that *x* instantiates *X.* The proposition $\lambda w \lambda t\, [X_{wt}\, x]$ has a truth-value gap. For instance, the property of having stopped smoking has a pre-requisite of being an ex-smoker. If somebody never smoked they could not stop smoking, of course. Then, however, the Composition $[^0True_{wt}\, \lambda w \lambda t\, [X_{wt}\, x]]$ is simply false and since it is an antecedent of the above implication, the implication is true, as it should be.

Since the topic of this paper is learning and refining concepts, we need to define the notion of concept. In TIL *concepts* are explicated as closed constructions in their normal form. Referring for details to [7, §2.2], we briefly recapitulate. Concepts are meanings of semantically complete terms that do not contain indexicals or other pragmatically incomplete terms. In case of the latter we furnish a pragmatically incomplete expression with an open construction containing free variables. An open construction cannot be executed unless valuation of its free variables is supplied, usually by the situation of utterance. For instance, the meaning of the sentence "He is smart" is the open construction $\lambda w \lambda t\, [^0Smart_{wt}\, he]$, $he \rightarrow \iota$, that cannot be evaluated until an individual is assigned to the free variable *he* as its valuation.[4] Hence, we don't treat this open

[3] For details see [7, §4.1]

[4] If such a sentence occurs in a broader discourse, its meaning can be completed by anaphoric references as well. For instance, in "John is a student, he is smart" the meanings are not pragmatically incomplete, because the individual John is substituted for the anaphoric variable *he.* For details on resolving anaphoric references in TIL, see [6]

construction as a concept. Since concepts should be at least in principle executable in any state of affairs, we explicate them as closed constructions.

However, our TIL constructions are a bit too fine-grained from the procedural point of view. Some closed constructions differ so slightly that they are *virtually* identical. In a natural language we cannot even render their distinctness, which is caused by the role of λ-bound variables that lack a counterpart in natural languages. These considerations motivated definition of the relation of *procedural isomorphism* on TIL constructions.[5] Procedurally isomorphic constructions form an equivalence class at which we can vote for a representative. To this end a normalization procedure has been defined that results in the unique *normal form C* of a construction that is a representative of the class of procedurally isomorphic constructions. Hence, we adopt this definition:

**Definition (*concept*)** A *concept* is a closed construction in its normal form.

For the sake of simplicity, in what follows we deal with concepts simply as with closed constructions, ignoring the above technicalities, because we believe that this simplification is harmless for our purposes.

The last notion we need to define is that of *refinement* of a concept. Basically, by refining a simple concept ${}^0O$ of an object $O$ we mean replacing ${}^0O$ by an equivalent molecular concept $D$ that produces the same object $O$. We also say that the molecular construction $D$ is an *ontological definition* of the object $O$.

Here is an example. The Trivialization ${}^0Prime$ is in fact the least informative procedure for producing the set of prime numbers. Using particular definitions of the set of primes, we can refine the simple concept ${}^0Prime$ in many ways, including:[6]

$$\lambda x\, [{}^0Card\ \lambda y\, [{}^0Divide\ y\ x] = {}^02],$$

$$\lambda x\, [[x \neq {}^01] \wedge \forall y\, [[{}^0Divide\ y\ x] \supset [[y = {}^01] \vee [y = x]]]],$$

$$\lambda x\, [[x > {}^01] \wedge \neg\exists y\, [[y > {}^01] \wedge [y < x] \wedge [{}^0Divide\ y\ x]].$$

The involved types are: ν, the type of natural numbers; *Card*/(ν(oν)): the cardinality of a set of natural numbers; *Divide*/(oνν): the relation of $x$ being divisible by $y$; the other types are obvious.

Thus, we define.

**Definition (*refinement of a construction*)** Let $C_1$, $C_2$, $C_3$ be constructions. Let ${}^0X$ be a simple concept of an object $X$ and let ${}^0X$ occur as a constituent of $C_1$. If $C_2$ differs from $C_1$ only by containing in lieu of ${}^0X$ an ontological definition of $X$, then $C_2$ is a *refinement of* $C_1$. If $C_3$ is a refinement of $C_2$ and $C_2$ is a refinement of $C_1$, then $C_3$ is a *refinement of* $C_1$.

*Corollary.* If $C_2$ is a refinement of $C_1$, then $C_1$, $C_2$ are equivalent but *not* procedurally isomorphic.

For instance, the simple concept of primes is not procedurally isomorphic with the above refinements, of course, which are molecular concepts with much richer structure than just ${}^0Prime$. As a result, the term 'prime' is not synonymous with its equivalents like 'the

[5] For details, see [4].

[6] For the sake of simplicity, here we again use infix notation without Trivialization for application of the binary relations >, < and the identity = between numbers.

set of naturals with just two factors', 'the set of naturals distinct from 1 that are divisible just by the number 1 and themselves', because the meanings of *synonymous terms* are procedurally isomorphic. Rather, 'prime' is only equivalent to these definitions.

So much for our formalism and background theory.

## 3. Supervised Machine Learning

Supervised machine learning is a method of predicting functional dependencies between input values and the output value. The supervisor provides an agent/learner with a set of training data. These data describe an object by a set of attribute values such that there is a functional dependency between these values.

For instance, a house can be characterized by its size, locality, date of building, architecture style, etc., and its price. Obviously, the price of a house depends on its size, locality, date of building and architecture style. Hence, the price is called an output attribute and the other attributes are input attributes. The goal of learning is to discover this functional dependency on the grounds of training data examples so that the agent can predict the value of the output attribute given the values of input attributes of a new instance.

More generally, where $x_1,\ldots,x_n$ are values of input attributes and $y$ an output attribute value, there is a function $f$ such that $y = f(x_1,\ldots,x_n)$. The goal of the learning process is to discover a function $h$ that approximates the function $f$ as close as possible. The function $h$ is called a *hypothesis*. The learner creates hypotheses on the grounds of training data (input-output values) provided by the supervisor. Correctness of the hypothesis is verified by using a set of test examples given their input attributes. The hypothesis is plausible if the learner predicts the values of the output attribute with a maximum accuracy.[7]

Since we decided to apply this method to learning *concepts*, we have to adjust the method a bit. First, instead of input/output attributes, we deal with concepts, that is closed constructions. The role of input 'attributes' is played by the constituents of a hypothetic molecular concept and instead of the output attribute we deal with the simple atomic concept that the learner aims to refine. The hypothetic function is that of a *requisite.* Training data are natural-language texts. The supervisor extracts from the text data positive and negative examples. For instance, let the 'output' concept to be learned be that of a cat, i.e. ${}^0Cat.$ The role of positive examples is played by particular descriptions of the property of being a cat like "Cat is a predatory mammal that has been domesticated". The learner establishes a hypothesis that the property

$$\lambda w \lambda t\, \lambda x\, [[[{}^0Predatory\ {}^0Mammal]_{wt}\, x] \wedge [{}^0Domesticated_{wt}\, x]]$$

belongs to the essence of the property *Cat.* Negative examples delineate the hypothesis from other similar objects. As a negative example for cat can serve the sentence "Dog is a domesticated predatory mammal that barks". This triggers a specialization of the hypothetic concept to the construction

$$\lambda w \lambda t\, \lambda x\, [[[{}^0Predatory\ {}^0Mammal]_{wt}\, x] \wedge [{}^0Domesticated_{wt}\, x] \wedge \neg[[{}^0Bark_{wt}\, x]]]$$

[7] For details, see [13], [15].

Hence, given a positive example, the learner refines the hypothetic molecular concept by adding other concepts to the essence, while a negative example triggers specialization of the hypotheses. The hypothetic concept can be also generalized. For instance, the learner can obtain as another positive example describing the property *Cat* the sentence "Cat is a *wild* feline predatory mammal". Since the properties *Wild* and *Domesticated* are inconsistent, the agent consults his/her ontology for a more general concept. If there is none, the 'union' of the properties, *Wild* or *Domesticated*, is included. As a result, the learner obtains this hypothesis.

$$\lambda w \lambda t\, \lambda x\, [[[^0\mathit{Feline}\ [^0\mathit{Predatory}\ ^0\mathit{Mammal}]]_{wt}\, x] \wedge [[^0\mathit{Domesticated}_{wt}\, x] \vee [^0\mathit{Wild}_{wt}\, x]] \wedge \neg[[^0\mathit{Bark}_{wt}\, x]]]$$

*Remark.* Both *Feline* and *Predatory* are property modifiers of type $((o\iota)_{\tau\omega}(o\iota)_{\tau\omega})$, i.e. functions that given an input property return another property as an output. Since these two modifiers are intersective, the rules of left- and right-subsectivity are applicable here.[8] In other words, predatory mammal is a predator and is a mammal, similarly for feline. If our agent has these pieces of information in their knowledge base, the above Composition $[[^0\mathit{Feline}\ [^0\mathit{Predatory}\ ^0\mathit{Mammal}]]_{wt}\, x]$ can be further refined to $[[^0\mathit{Feline}'_{wt}\, x] \wedge [^0\mathit{Predatory}'_{wt}\, x] \wedge [^0\mathit{Mammal}_{wt}\, x]]$, where *Feline*' and *Predatory*' are properties of individuals, i.e. objects of type $(o\iota)_{\tau\omega}$.

Both generalization, specialization and conjunctive extension are methods of refining a hypothetic concept, the methods that we are going to describe in the next section.

### *3.1. Refining hypothesis space*

In our method we try to find the description of all plausible hypotheses that are consistent with the training data and are derivable from the provided examples.[9] To this end we assume that there is no noise in the training data [13]. In other words, the examples provided to the learner are adequate for the prediction of the refined concept. Obviously, a learner can usually examine just a small finite training set of examples instead of a possibly infinite set of sample concepts. Hence, *inductive learning* is applied to obtain a hypothetic concept.[10] In the process of inductive learning, the relation '*more general*' defined on the set of hypotheses is used. This relation is defined as follows. Let $h_1$, $h_2$ be hypothetic concepts defined on an input domain *X.* Then $h_1$ is *more general* then $h_2$, in symbols '$h_2 \Rightarrow h_1$', iff

$$\forall x \in X\ [(h_2(x) = 1) \supset (h_1(x) = 1)].$$

*Note.* By $(h_i(x) = 1)$ we mean that an object *x* falls under the concept $h_i$ in a given state of affairs. Hence, this simplified notation can be read as "all objects *x* that fall under the concept $h_2$ fall also under the more general concept $h_1$".

[8] For details and analysis of other kinds of modifiers, see [5].

[9] Hypothesis is consistent with the training data, i.e. the set *S* of examples, if the value predicted by the hypothesis is the value of output attribute of all examples belonging to *S*.

[10] For details on and definition of inductive learning see, e.g., [13, §2.2.2, p. 23].

The subset of hypotheses obtained by inductive learning which is consistent with the training set of examples is called *version-space*.

### *3.2. Algorithm framework*

All machine learning algorithms, no matter into which family they belong, can be characterized by common categories which form a framework [10]. The algorithms are characterized by task goals, training data, data representation, and a set of operators which manipulate with data representation. In our machine learning algorithm, the framework can be briefly described as follows.

*Objective Goal*. As mentioned above, the goal of an agent is to discover the best refinement of the learned simple concept of an object *O*, i.e. a molecular closed construction that produces the same object. Moreover, this molecular concept should specify as much as possible of the requisites of the object *O* so that it also excludes other similar concepts.

*Training data*. An agent works with positive and negative examples that are sentences extracted by a supervisor from a textual base. Positive examples contain concepts of requisites specifying the learned simple concept, while negative examples specify properties that do not belong to the essence of the intension provided by the concept.

*Data Representation*. The agents must have an internal formal representation of data obtained by examples. Plausible hypotheses are then formulated in terms of this representation. Our formalism is that of Transparent Intensional Logic so that the sentences are analysed in terms of TIL constructions.

*Knowledge Modifying Module*. The learning algorithm is biased in favour of a preferred hypothesis. By using proper preferences, we reduce the hypothesis space. In version-space learning the bias is called a restriction bias, because the bias is obtained by restricting the allowable hypotheses. The agent uses a set of operations to modify the hypothesis during a *heuristic* search in the hypothesis space. The three main operations to modify a hypothetic concept are generalization, specialization and refinement. There are two possibilities how to obtain a proper hypothesis. The first one is based on using merely positive examples. In this case we need to be sure that the examples cover well the positive cases; in other words, we need examples containing all and only requisites of the learned concept. The second way that we vote for is using both positive and negative examples. By applying specialization based on negative examples we exclude too general hypotheses.

## 4. Inductive heuristics

For our purpose we voted for an adjusted version of Patrick Winston algorithm [17] of supervised machine learning. This algorithm applies the principles of generalization and specialization to obtain a plausible hypothesis, i.e. the functional dependency between input and output attributes. In our case the main principle is the method of refining the output simple concept. Hence, instead of a functional dependency between input and output attributes, we are looking for molecular concepts refining the output simple

concept the constituents of which are related to the output concept by the requisite relation. Winston algorithm assumes that examples differ from the model just in one attribute while in our case we develop the molecular concept by adding new constituents contained in example sentences describing or rather refining the output concept. Hence our algorithm does not compare a model with examples; rather, it compares the hypothetic concept with information in sample sentences.

As stated above, our main method is *refinement* of a concept, i.e. a hypothetic construction. Based on positive examples we extend the collection of requisites by adding missing concepts in a conjunctive way. As a special case, *generalization* can be applied. Based on agents' ontologies, generalization usually concerns replacing one or more constituents of the hypothetic concept by a more general one.

*Specialization* is triggered by negative examples. As a result, negation of a property that does not belong to the essence of the hypothetic concept is inserted. Specialization serves to distinguish the output concept from similar ones. For instance, a wooden horse can serve as a negative example to the concept of horse, because a wooden horse is not a horse; rather, it is a toy horse though it may look like a genuine living horse.

Heuristic methods of the original Winston algorithm work with examples that cover all the attributes of a learned object. Based on positive examples the hypothesis is modified in such a way that the values of attributes are adjusted, or in case of a negative example an unwanted attribute marked as Must-not-be is inserted. In our application the sentences that mention the learned concept contain as constituents some but not all the requisites of this concept, and we build up a new molecular concept by adding new information extracted from positive or negative examples. Hence, we had to implement a new heuristic *Concept-introduction* for adding concepts of new requisites into a hypothetic concept. Negative examples trigger the method *Forbid-link* that inserts a concept of negated property into the hypothesis. Generalization is realized by modules that introduce a concept of a more general property; to this end we also adjusted the original heuristic *Close-interval* so that it is possible to generalize values of numeric concepts by the union of interval values from an example and model.[11]

Here is a brief specification of the algorithm.

*Refinement.*

1. Compare the model hypothesis (to be refined) and the positive example to find a significant difference
2. If there is a significant difference**,** then
   a) if the positive example contains as its constituent a concept that the model does not have, use the ***Concept-introduction***
   b) else ignore example

*Specialization*.

1. Compare the model hypothesis (to be refined) and the near-miss example to find a significant difference
2. If there is a significant difference**,** then

[11] For the sake of simplicity, we did not change the original names of particular modules though we do not work with 'links' between objects and attribute values any more. The heuristics *Require-link* and *Drop-link* from the original algorithm have not been used in our adjusted version.

c) if the near-miss example has a constituent of the concept that the model does not have, use the ***Forbid-link***
d) else ignore example

*Generalization.*
1. Compare the model hypothesis (to be refined) and the positive example to determine a difference
2. For each difference do
   a) if a concept in the model points at a value that differs from the value in the example, then
      i) if the properties in which the model and example differ have the most specific general property, use the ***Climb-tree***
      ii) else use ***Union-set***
   b) if the model and example differ at an attribute numerical value or interval, use the ***Close-interval***
   c) else ignore example.

## 5. Example of learning the concept of myopia

As a sample example we now introduce the process of learning refinements of the simple concept of myopia, i.e. ${}^0$*Myopia*, by extracting information from natural language sentences describing the property of having myopia.

As always, first types.

*Myopia* / $(o\iota)_{\tau\omega}$
*Sharp, Blur, Disorder, Eye_nerve, Eye_lenses* / $(o\iota)_{\tau\omega}$
*Eye_focus, Damaged, Inflexible* / $((o\iota)_{\tau\omega}(o\iota)_{\tau\omega})$
*Close, Distant, Looking_at* / $(o\iota\iota)_{\tau\omega}$
$x, y \rightarrow \iota$
*Req* / $(o(o\iota)_{\tau\omega}(o\iota)_{\tau\omega})$
$\exists$ / $\big(o(o\iota)\big)$

***Positive examples***:

1. In myopia, close objects look sharp.

$$\left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ \exists \lambda y \left[ \left[ \left[ {}^0Looking\_at_{wt}\ x\ y \right] \wedge \left[ {}^0Close_{wt}\ x\ y \right] \right] \supset \left[ {}^0Sharp_{wt}\ y \right] \right] \right] {}^0Myopia \right] \right]$$

2. In myopia, distant objects appear blurred.

$$\left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ \exists \lambda y \left[ \left[ \left[ {}^0Looking\_at_{wt}\ x\ y \right] \wedge \left[ {}^0Distant_{wt}\ x\ y \right] \right] \supset \left[ {}^0Blur_{wt}\ y \right] \right] \right] {}^0Myopia \right] \right]$$

3. It is an eye focusing disorder.

$$\left[ {}^0Req \left[ {}^0Eye\_focus\ \ {}^0Disorder \right] {}^0Myopia \right]$$

***Negative examples***.

1. Cause of myopia is not damaged eye-nerve.

$$\neg \left[ {}^0Req \left[ {}^0Damaged\ \ {}^0Eye\_nerve \right] {}^0Myopia \right]$$

2. Cause of myopia is not inflexible eye lenses.

$$\neg \left[ {}^0Req \left[ {}^0Inflexible\ \ {}^0Eye\_lenses \right] {}^0Myopia \right]$$

***Simulation of the algorithm execution***.

The execution of our algorithm begins with a first chosen positive example. The construction encoded by this sentence becomes an initial model.

"*In myopia, close objects look sharp.*"

$$\left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ \exists \lambda y \left[ \left[ \left[ {}^0Looking\_at_{wt}\ x\ y \right] \wedge \left[ {}^0Close_{wt}\ x\ y \right] \right] \supset \left[ {}^0Sharp_{wt}\ y \right] \right] \right] {}^0Myopia \right] \right]$$

The second positive example

"*In myopia distant objects appear blur.*"

refines the model by *Concept-introduction*. As a result, we have a hypothetic model "*In myopia, close objects look sharp and distant objects look blur*".

$$\left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ \left[ \exists \lambda y \left[ \left[ \left[ {}^0Looking\_at_{wt}\ x\ y \right] \wedge \left[ {}^0Close_{wt}\ x\ y \right] \right] \supset \left[ {}^0Sharp_{wt}\ y \right] \right] \right] \right. \right.$$
$$\wedge \left[ \exists \lambda y \left[ \left[ \left[ {}^0Looking\_at_{wt}\ x\ y \right] \wedge \left[ {}^0Distant_{wt}\ x\ y \right] \right] \right. \right.$$
$$\left. \left. \left. \left. \supset \left[ {}^0Blur_{wt}\ y \right] \right] \right] \right] \left. \left. {}^0Myopia \right] \right]$$

The last positive example

"*It is an eye focusing disorder.*"

also refines the model by *Concept-introduction*:

$$\left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ \left[ \exists \lambda y \left[ \left[ \left[ {}^0\text{Looking\_at}_{\text{wt}}\ \text{x}\ \text{y} \right] \wedge \left[ {}^0\text{Close}_{\text{wt}}\ \text{x}\ \text{y} \right] \right] \supset \left[ {}^0\text{Sharp}_{\text{wt}}\ \text{y} \right] \right] \right] \right. \right.$$
$$\wedge \left[ \exists \lambda z \left[ \left[ \left[ {}^0Looking\_at_{wt}\ x\ z \right] \wedge \left[ {}^0Distant_{wt} x\ z \right] \right] \right. \right.$$
$$\left. \left. \left. \supset \left[ {}^0Blur_{wt}\ z \right] \right] \wedge \left[ \left[ {}^0Eye\_focus\ \ {}^0Disorder \right]_{wt} x \right] \right] \left. {}^0Myopia \right] \right]$$

The first negative example "The cause of myopia is not a damaged eye nerve" triggers specialization of the hypothesis. As a result, we apply the negated property that is added into the essence of myopia:

$$\left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ \left[ \exists \lambda y \left[ \left[ \left[ {}^0Looking\_at_{wt}\ x\ y \right] \wedge \left[ {}^0Close_{wt}\ x\ y \right] \right] \supset \left[ {}^0Sharp_{wt}\ y \right] \right] \right] \right. \right.$$
$$\wedge \left[ \exists \lambda z \left[ \left[ \left[ {}^0Looking\_at_{wt}\ x\ z \right] \wedge \left[ {}^0Distant_{wt}\ x\ z \right] \right] \right. \right.$$
$$\left. \left. \supset \left[ {}^0Blur_{wt}\ z \right] \right] \wedge \left[ \left[ {}^0Eye\_focus\ \ {}^0Disorder \right]_{wt} x \right] \right.$$
$$\left. \left. \wedge \left[ {}^0\neg \left[ {}^0Damaged\ \ {}^0Eye\_Nerve \right]_{wt} x \right] \right] \left. {}^0Myopia \right] \right]$$

The second negative example "Myopia is not caused by inflexible eye lenses" also specializes the concept. The resulting molecular concept defining the property of myopia is this:

$$\left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ \left[ \exists \lambda y \left[ \left[ \left[ {}^0Looking\_at_{wt}\ x\ y \right] \wedge \left[ {}^0Close_{wt} x\ y \right] \right] \supset \left[ {}^0Sharp_{wt}\ y \right] \right] \right] \right. \right.$$
$$\wedge \left[ \exists \lambda z \left[ \left[ \left[ {}^0Looking\_at_{wt}\ x\ z \right] \wedge \left[ {}^0Distant_{wt}\ z\ y \right] \right] \right. \right.$$
$$\left. \left. \supset \left[ {}^0Blur_{wt}\ z \right] \right] \right] \wedge \left[ \left[ {}^0Eye\_focus\ \ {}^0Disorder \right]_{wt} x \right]$$
$$\wedge \left[ {}^0\neg \left[ {}^0Damaged\ {}^0Eye\_Nerve \right]_{wt} x \right]$$
$$\left. \left. \wedge \left[ {}^0\neg \left[ {}^0Inflexible\ \ {}^0Eye\_lenses \right] x \right] \right] \right] {}^0Myopia \Bigg]$$

In this example we did not deal with generalization. It might concern calling the *Close-interval* module or dealing with subsective modifiers of the property of being shortsighted. For instance, in Wikipedia we can read:

The degree of myopia is described in terms of the power of the ideal correction, which is measured in dioptres:

- Low myopia usually describes myopia of −3.00 dioptres or less (i.e. closer to 0.00).
- Moderate myopia usually describes myopia between −3.00 and −6.00 dioptres.
- High myopia usually describes myopia of −6.00 or more.

By the analysis of these sentences we would insert into the definition of myopia other three concepts defining myopia, namely low, moderate and high myopia. By applying generalization, we obtain still another definition, namely that myopia is measured in negative dioptres.

## 6. Conclusion

In this paper we introduced the basic principles of supervised machine learning, namely the method of refining hypothesis by means of positive and negative examples. The process of refinement of a given hypothesis triggered by positive and near-miss examples together with heuristic functions that modify the hypothesis has been described and illustrated by examples. We applied an adjusted version of Patrick Winston's data driven algorithm for machine learning. The area under scrutiny has been agents' learning simple concepts by their refining. In other words, our agents learn new concepts by discovering compound definitions of the objects specified just by a simple concept of Trivialization. The method itself has been illustrated by the example of agent's learning the concept of myopia. Our data have been formalized by means of the TIL tools, namely constructions and types produced by the NLA algorithm [11].

The proposed machine learning method heavily relies on the role of a supervisor. For a success in learning it is important that the supervisor extracts from a given text those sentences that mention the concept in a way plausible for learning. Moreover, there should not by any noise in these input data, and the supervisor should properly classify these sentences into positive and negative examples. Hence, we assume that the role of

a supervisor is played by an experienced linguist. As a future research, we intend to extend the functionalities of the algorithm so that it will cover also the extraction of sample sentences where the output learned concept receives mention. Though there is no substitute for a supervisor in a supervised machine learning method, its role can be at least partly played by an algorithm so that the manual work of a linguist is reduced to a minimum. Our next goal is to improve the method so that the agents would learn synonymous terms referring to the same concept as well and distinguish them from merely equivalent ones. This is important for dealing with hyperintensional attitudes of knowing, believing, designing, calculating, solvind, etc. properly. These attitudinal verbs are part and parcel of our everyday vernacular so that their proper analysis and logic should not be missing from any automatized multiagent system. And since these attitudinal verbs establish hyperintensional contexts where the substitution of merely equivalent terms fails, the agents need to know the synonyms of the learned concepts as well.

**Acknowledgements**. This research has been supported by the Grant Agency of the Czech Republic, project No. GA18-23891S, "Hyperintensional Reasoning over Natural Language Texts" and also by the internal grant agency of VSB-Technical University Ostrava, project No. SP2019/40, "Application of Formal Methods in Knowledge Modelling and Software Engineering II".

## References

1. Číhalová, M., Duží, M., Menšík, M. (2014). Logical specification of processes. *Frontiers in Artificial Intelligence and Applications*, vol. 260: Information Modelling and Knowledge Bases XXV, IOS Press, 45-63.
2. Duží, M. (2012). Extensional logic of hyperintensions. *Lecture Notes in Computer Science*, vol. 7260, pp. 268-290. DOI: 10.1007/978-3-642-28279-9-19
3. Duží, M. (2014). Communication in a multi-cultural world. *Organon F*, vol. 21, No. 2, pp. 198-218.
4. Duží, M. (2017). If structured propositions are logical procedures then how are procedures individuated? *Synthese* special issue on the Unity of propositions. DOI: 10.1007/s11229-017-1595-5
5. Duží, M. (2017). Property modifiers and intensional essentialism. *Computación y Sistemas,* vol. 21, No. 4, 2017, pp. 601–613. DOI: 10.13053/CyS-21-4-2811.
6. Duží, M. (2018). Logic of Dynamic Discourse; Anaphora Resolution. *Frontiers in Artificial Intelligence and Applications*, vol. 301: Information Modelling and Knowledge Bases XXIX, pp. 263-279, Amsterdam: IOS Press, DOI 10.3233/978-1-61499-834-1-263
7. Duží, M., Jespersen, B., Materna, P. (2010). *Procedural Semantics for Hyperintensional Logic. Foundations and Applications of Transparent Intensional Logic*. Berlin: Springer.
8. Francez, N. (2015). *Proof-theoretic Semantics.* Studies in Logic 57, College Publications.
9. Kovář, V. Baisa, V, Jakubíček, M. (2016). Sketch Engine for Bilingual Lexicography. International *Journal of Lexicography*, vol. 29, No. 3, pp. 339-352.

10. Luger G. F. (2009). *Artificial intelligence: structures and strategies for complex problem solving*. 6th ed. Boston: Pearson Addison-Wesley, 2009. ISBN 978-0-321-54589-3.
11. Medveď, M., Šulganová, T. Horák, A. (2017). Multilinguality Adaptations of Natural Language Logical Analyzer. In Proceedings of the 11th Workshop on *Recent Advances in Slavonic Natural Language Processing, RASLAN 2017*, Brno: Tribun EU, pp. 51-58.
12. Menšík, M., Duží, M., Albert, A., Patschka, V., Pajr, M. (2019). Machine learning using TIL. To appear in Proceedings of *the 29th International Conference on Information Modelling and Knowledge Bases, EJC 2019*.
13. Mitchell T. M. (1997). *Machine Learning*. New York: McGraw-Hill, 1997. ISBN 00-704-2807-7.
14. Poole D. L., Mackworth A. K. (2010). *Artificial intelligence: foundations of computational agents*. 2nd pub. Cambridge: Cambridge University Press, 2010. ISBN 978-0-521-51900-7.
15. Russell S. J., Norvig P. (2014). *Artificial intelligence: a modern approach*. 2nd ed. Harlow: Pearson Education, 2014. ISBN 978-1-29202-420-2.
16. Tichý, P. (1988). *The Foundations of Frege's Logic.* Berlin, New York: de Gruyter.
17. Winston P. H. (1992). *Artificial intelligence*. 3rd ed., Mass.: Addison-Wesley Pub. Co., 1992. ISBN 02-015-3377-4.