Intuitive representation of computable knowledge

doi.org/10.20944/preprints202007.0486.v2

Received: 29 July 2020

Check for updates

Steven Vercruysse [1,2,3]* & Martin Kuiper [1,3]

Scientific progress is increasingly dependent on knowledge in computation-ready forms[1–9]. In the life sciences, among others, many scientists therefore extract and structure knowledge from the literature[1,3,10–19]. In a process called manual curation, they enter knowledge into spreadsheets, or into databases where it serves their and many others' research. Valuable as these curation efforts are, the range and detail of what can practically be captured and shared remains limited, because of the constraints of current curation tools. Many important contextual aspects of observations described in literature simply do not fit in the form defined by these tools, and thus cannot be captured[14,15]. Here we present the design of an easy-to-use, general-purpose method and interface, that enables the precise semantic capture of virtually unlimited types of information and details, using only a minimal set of building blocks. Scientists from any discipline can use this to convert any complex knowledge into a form that is easily readable and meaningful for both humans and computers. The method VSM forms a universal and high-level language for encoding ideas, and for interacting with digital knowledge.

The ability to extend our knowledge by building on the results of others is crucial to the progress of science. Yet, information dissemination in science happens mainly via publications, expressed as stories with tables and figures. This method of scientific discourse is extremely flexible and expressive for disseminating new findings, but it makes them essentially only available by reading. This is highly time-consuming for scientists who may need to read and extract bits of information from thousands of relevant texts.

Initiatives such as FAIR, nano- and micropublications are already guiding scientists beyond this purely narrative way of sharing results[20–24]. FAIR encourages us to make scientific data 'Findable, Accessible, Interoperable, Reusable', and overall better human- and machine-interpretable. Although FAIR is often mentioned with a focus on the reuse of experimental data, FAIR's principles apply equally to the new insights derived from that data, as reported in publications. But unlike *data*, which is sufficiently uniform to fit into database schemas, the new insights and their detailed context (together called *information* or *knowledge*) are highly diversiform and thus not easy to capture with existing technology. Yet for maximum benefit of computational processing, scientific knowledge locked up as unstructured text in publications needs to be mobilized into machine-interpretable form[12,25].

Modeling the world

Such *computable knowledge* is becoming a cornerstone of today's science, especially for integrative research on how numerous interacting components work together as a system. For instance, computers assist researchers who piece together the structure of brain neuronal networks, as a basis to understand function[1]. Other computer models simulate cell-fate decision networks in cancer, enabling biologists to study molecular system dynamics, predict responses to treatments, and prioritize which lab experiments to carry out next[2]. Others assess ecosystem changes[4], genetic diseases across species[5], transmission of ideas in social media[7], or financial activity networks[9]. Such models must be constructed from a sufficiently detailed set of prior knowledge in machine-actionable form.

Scientists around the globe are therefore gathering information about components and their relationships from the scientific literature, and translate them into structured, computable forms[10,11]. This process is called *curation* and is performed both institutionally[13,15,16] and in many individual research projects[1,3,11,17,18]. It happens largely manually or semi-automatically at best, as human understanding remains essential to properly interpret a publication's context and details on observations. Synthesizing text into reliable assertions about a component's function is no obvious task for machines alone[26,27].

The struggle with curation tools

There has never existed a multi-purpose, practical tool that enables people to easily capture and structure all relevant information from a piece of text. Much of this information can be highly relevant for building models, but it can be extremely diverse and irregular. Publications report on diverse information types (fields, subjects), and with any variety or amount of contextual details (exper-

[1]Systems Biology group, Dept. of Biology, Norwegian University of Science and Technology, Trondheim, Norway. [2]Independent Scientist. [3]These authors share senior authorship.
e-mail: vercruys@alumni.ntnu.no; martin.kuiper@ntnu.no. *Corresp. author.
Keywords: knowledge representation, curation, biocuration, semantics, systems biology, ontology, user interface, VSM.

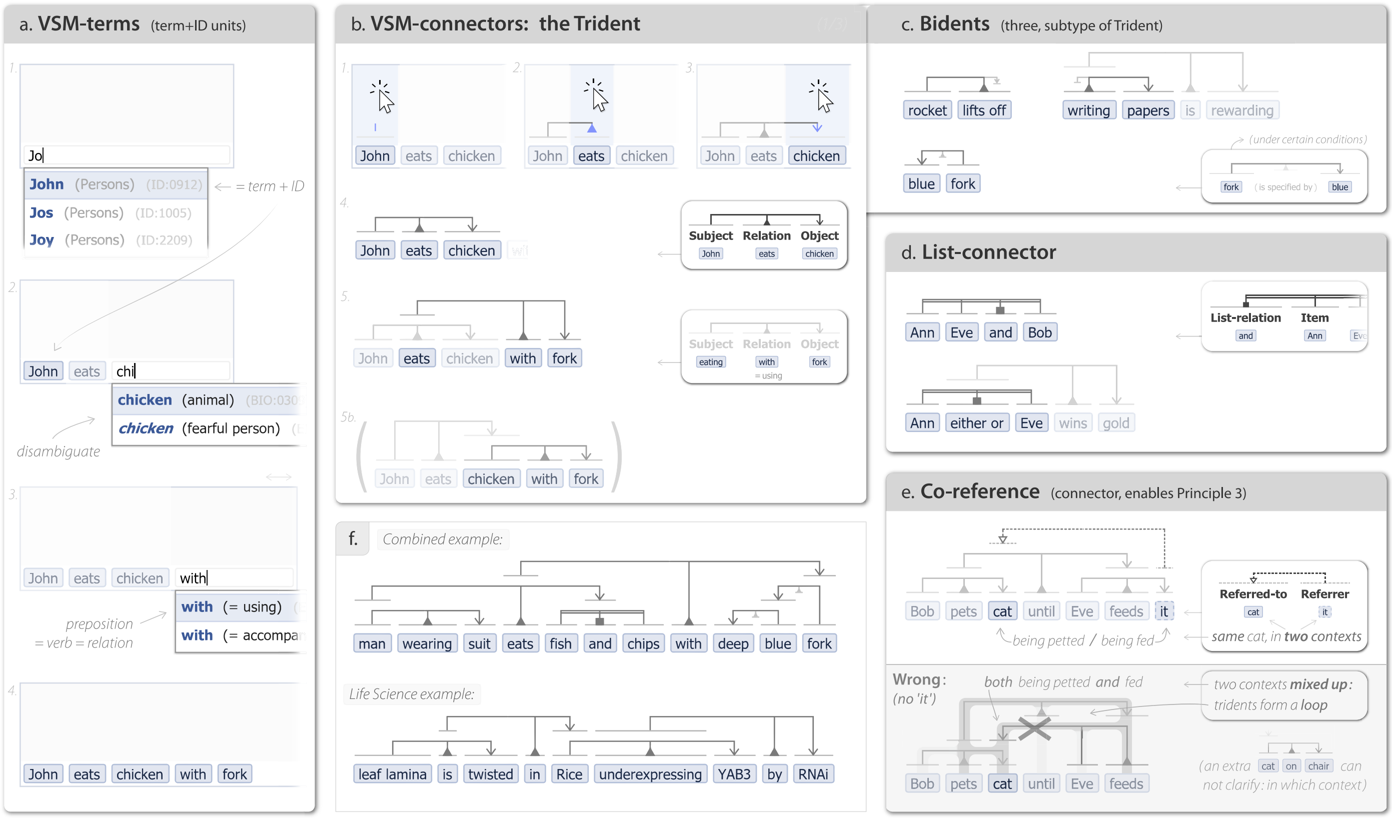

a. VSM-terms (term+ID units)
1.
Jo
John (Persons) (ID:0912)
= term + ID
Jos (Persons) (ID:1005)
Joy (Persons) (ID:2209)
2.
John eats chi
chicken (animal) (BIO:030
chicken (fearful person)
disambiguate
3.
John eats chicken with
with (= using)
with (= accompa
preposition
= verb = relation
4.
John eats chicken with fork
b. VSM-connectors: the Trident
1. John eats chicken
2. John eats chicken
3. John eats chicken
4. John eats chicken
Subject Relation Object
John eats chicken
5. John eats chicken with fork
Subject Relation Object
eating with fork
= using
5b. John eats chicken with fork
c. Bidents (three, subtype of Trident)
rocket lifts off
writing papers is rewarding
blue fork
(under certain conditions)
fork (is specified by) blue
d. List-connector
Ann Eve and Bob
List-relation Item
and Ann Eve
Ann either or Eve wins gold
e. Co-reference (connector, enables Principle 3)
Bob pets cat until Eve feeds it
being petted / being fed
Referred-to Referrer
cat it
same cat, in two contexts
Wrong: (no 'it')
both being petted and fed
two contexts mixed up: tridents form a loop
Bob pets cat until Eve feeds
(an extra cat on chair can not clarify: in which context)
f. Combined example:
man wearing suit eats fish and chips with deep blue fork
Life Science example:
leaf lamina is twisted in Rice underexpressing YAB3 by RNAi

Fig. 1 | Main elements of VSM: precise, intuitive semantics; clear interface. **a**, A user searches for terms in linked dictionaries (three of five searches shown) in a *vsm-box*. Each autocomplete panel item represents a term linked to a particular identifier (ID); when selected, their combination appears as a *VSM-term*. By choosing particular word-forms, an easily readable sequence of VSM-terms is created. **b**, The user indicates units of three VSM-terms that relate to each other as subject, relation, and object, by adding *tridents* (a type of *VSM-connector*). A first trident is created via clicks above three VSM-terms. Any additional trident connects a VSM-term to further details about it. Different connection structures create different meanings for a same term set. **c**, Any trident-*leg* may be omitted to create a *bident* (three kinds, trident subtypes). The *relation-omitting bident* implies an implicit relation 'is specified by', and may only be valid if ontological reasoning can infer a more precise relation, like 'has color'. The availability of bidents illustrates VSM's flexibility and focus on usability. **d**, The *list-connector* combines *list-items* based on some *list-relation*. **e**, The *co-reference* and its functionality. First, it supports the use of referring terms like `it`. Here it clarifies that `it` and `cat` represent the same cat. Second, this makes VSM-sentences semantically correct: `cat` and `it` represent that cat in two different situations or *local contexts*; this enables adding details about the cat in each separate context, see Fig. 3b. If connectors (excl. coreferences) form a loop, it indicates that conflated, ambiguous context is present, and a coreference is needed; see Principle 3. Third, coreferences may refer to terms in earlier VSM-sentences, when building a story (e.g. experiment protocol). **f**, One can repeatedly add further context details to any VSM-term by connecting more VSM-terms, to capture knowledge of any shape or context-richness.

iment setup, conditions, etc.). Still, the more context one can record, the more useful the collected knowledge becomes, as one can take into account the conditions under which it is valid.

Current curation tools are often built around predefined entry forms (or just spreadsheet tables), covering only what one anticipates to capture. However, while reading papers, scientists typically discover extra types of information or relevant details that should be captured too. Yet doing so often means managing an ever-widening spreadsheet, or requires laborious updates to the database and user-interface; and therefore happens only infrequently or improvisedly[14,15]. This condemns possibly essential context and associated knowledge to remain buried in unstructured text and inaccessible for computational use. Some curation tools are more flexible, but these are hard to use. Controlled languages[28,29] involve daunting grammars that users need to learn, and current semantic technologies[7,30] are only usable by experts in knowledge representation (KR), not by typical biologists, chemists, etc.

This also prevents curation from scaling up. Professional curators are scarce, so some form of crowdsourcing that involves many more scientists will be needed to curate much of the relevant literature. This implies that we will only be able to leverage the full potential of computational processing on the vast archives of humanity's research results, after we have curation technology that is flexible and intuitive enough to be used widely.

Solution

We aimed to design a new approach that is both extremely flexible for capturing diverse and rich information, and equally focuses on being intuitive to understand and use.

VSM (Visual Syntax Method) is a new semantic model, combined with a supporting user-interface (UI). VSM enables one to manually reformulate any unit of information, into a clear, precise semantic form, whereby any inherent complexity is kept manageable. VSM evolved from discussions with scientists from diverse domains over ten years.

Here we describe the design of both this method and its supporting UI. Applications, software, and links to existing technology are detailed in separate papers[31–34].

Available material

The main text introduces VSM to an audience across scientific disciplines. It starts with domain-neutral examples, allowing readers

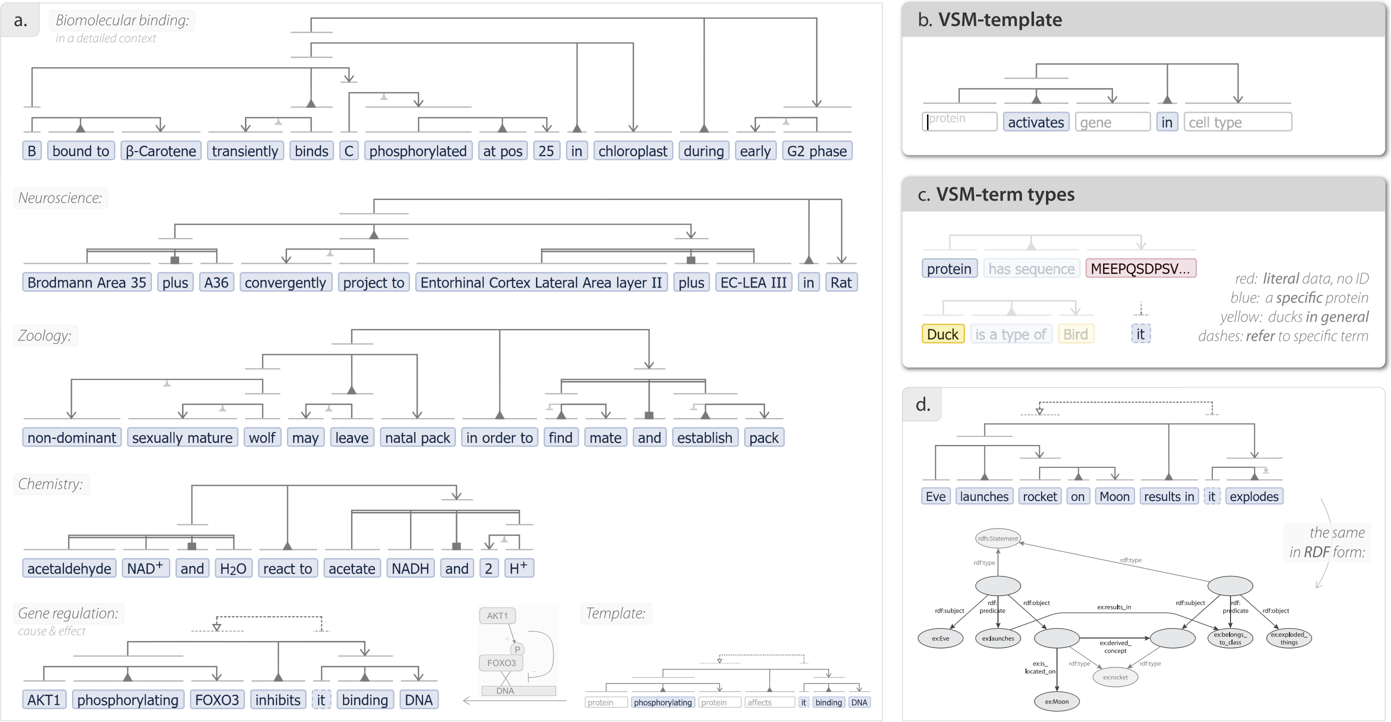


Fig. 2 | Cross-disciplinary applicability and intuitiveness of VSM. **a**, VSM-sentences can represent information from diverse research domains, into one unified, human-readable, computable language. The examples assume the availability of dictionaries (=controlled vocabularies, CVs) that provide a term+ID for each shown VSM-term. They also assume the availability of synonyms (terms linked to a same ID) that help make VSM-sentences more readable like natural language: e.g. prepositions for verbs (in, is located in), conjugations (bind/-s/-ing), adverbs (convergent/-ly), or abbreviations (A36). These word-form variants could be provided directly by the CV or be generated by intermediary code when needed. Also numeric concepts (numbers, dates) can be generated ad-hoc[31]. While CVs are always work-in-progress, here we focus on how VSM brings such terms together, syntactically and semantically. The bottom example shows a template, filled out with biological observations. **b**, VSM-templates are partially completed VSM-sentences that mimic form-based input. They facilitate larger curation efforts focused on capturing specific types of knowledge from literature. Templates include VSM's other advantages: autocomplete (faster term lookup, lower error rates), the flexibility to connect additional VSM-terms as needed, and the ability to present information from diverse curation projects into one multi-purpose user-interface. **c**, VSM-term types. *Instance*: concept within a specific context (blue; default and predominant); *referring instance*: refers to another instance; *class*: concept that is not context-bound, general category (yellow; e.g. for ontological 'x is-a y' hierarchies); *literal*: text or data without ID (red; e.g. protein sequence data). **d**, A VSM-sentence was manually converted to a possible graph representation in RDF, a widely used knowledge representation form[30]. For most people, the VSM-sentence is much easier to understand than RDF. See Supplementary Information Fig. S10 for a detailed comparison of what makes VSM structurally more elegant.

to focus on basic principles, and then shows how any particular curation need may be served. Supplementary Information provides extra detail, justifies design choices toward KR experts, and extends the Discussion. Online pages[31] provide many editable examples for hands-on experience with VSM.

VSM as a Semantic Model

VSM is like a language: it provides a small set of elements for constructing information as *VSM-sentences*. These are flexible, semi-linear statements. Similar to how any language consist of words and a grammar, VSM consists of *VSM-terms* and *VSM-connectors* (Fig. 1), with rules to combine these meaningfully.

VSM includes certain features from existing methods, including natural language, controlled languages[28,29], ontologies[9,13,35], and RDF (Resource Description Framework)[30]. But VSM's combination of high usability, flexibility, and semantic preciseness for computability also requires a fresh set of foundational principles.

VSM as a User Interface

VSM includes the design of a user-interface for entering or reading VSM-sentences. This *VSM-box* input-component holds one VSM-sentence (Fig. 1) and should be embedded in other software, like a web page[32].

A VSM-box should be connected to one or more *dictionary* resources that provide terminology. For instance it could be linked to terms from the English dictionary, lists of protein or gene names and IDs, and ontologies from resources like BioPortal[33,35–37]. Each dictionary is a list of *concepts* within a particular domain. A concept is the combination of a computer-processable, unique *identifier* (ID), and one or more human-readable *terms* (synonyms); whereby terms can consist of several words, e.g. 'is located in'.

VSM-terms

We will use a VSM-box to introduce VSM's design. In Fig. 1a, a VSM-box is linked to a set of dictionaries that together provide the terms and IDs necessary for composing a VSM-sentence.

A user enters the observation that John eats a chicken with a fork. While typing in a VSM-box, an autocomplete panel brings up matching terms from the linked dictionaries. The panel shows distinguishing information about each term (e.g. description, dictionary, ID). The user repeatedly chooses items from these panels, and thereby creates five *VSM-terms*: these are UI-elements that represent a term plus a linked ID.

Notice that by combining multiple dictionaries, some terms may refer to multiple IDs (e.g. a gene-name referring to genes from multiple species), and multiple synonymous terms may refer to a same

ID (e.g. a gene having multiple names). The autocomplete helps users disambiguate this, and while the resulting VSM-term may show only a term, the linked ID is stored underneath.

VSM's support for *synonyms* also enables VSM-sentences to look more like natural language, which enhances intuitiveness[38]. In Fig. 1a, the preposition `with` would represent the same relation as the verb `uses`: they are synonymous word-forms linked to the same ID. This allows the sequence of VSM-terms `John eats chicken with fork` to be easier to understand for people than an equivalent `John eats chicken uses fork`.

VSM-connectors

Next, the user specifies a syntactic structure among VSM-terms by adding *VSM-connectors*. These come in just three main types, the primary being the *trident*.

Tridents visually organize terms into *triples*, which are units of three VSM-terms that relate to each other as Subject, Relation, and Object. Triples are one of the intuitive, basic units of conceptualization[30,39] used in VSM. The user specifies a triple by clicking above three terms in said order. This adds a trident connector, attaching to each term with a distinctly drawn *leg* (plain, up-triangle, down-arrow, resp.) representing the term's assigned role in that triple.

The first trident in Fig. 1b specifies the unit `John eats chicken`. The second trident again connects three *individual VSM-terms*, specifying a second unit that can now be read as: 'the eating (of chicken by John)' (=subject) happens 'using' (=relation) 'a fork' (=object). Notice that we here refer to the verb 'eats' via an equivalent noun: an 'eating' activity. Another version where 'a chicken uses a fork' shows that attaching connectors to different terms changes the whole meaning.

VSM Principles

The example introduces a way of thinking that can be scaled up to knowledge of any complexity. This is defined in three principles for how to build or interpret a VSM-sentence. These are presented from an end-user's perspective. Mathematical (formal-logic etc.) perspectives are discussed elsewhere[31]. The principles are introduced below, elaborated upon in Supplementary Information, and summarized in Fig. 4.

Principle 1. *Bottom-up construction.* Although a VSM-sentence may resemble natural language, it is not. It is a condensed piece of information, reformulated by a curator in such a way that VSM-terms can be grouped with VSM-connectors. Connectors are not placed on top of unstructured text; instead they assemble a new VSM-sentence via meaningful links between individual VSM-terms. VSM-term labels display information readably, IDs define meanings precisely, and connectors define underlying conceptual structure clearly.

Principle 2. *Referable entities.* VSM-terms may appear as verbs, adjectives, etc., but only appear in that word-form to make a VSM-sentence more human-readable. For example, a term like 'eats' (=verb-form) may be given the relation-role under one trident, but one can always refer to it again and give it a subject- or object-role (i.e. an entity-role) under another connecting trident, where one would rather call it 'the eating' (=noun-form) (Fig. 1b). This means that in VSM, 'traditional relations' and entities are treated equally; and actually *every* VSM-term is a referable *entity* to which additional VSM-terms can be connected. Every verb, etc. term may therefore have a synonymous noun (with same ID), e.g. blue/blueness. One may even think of all VSM-terms as nouns, to keep in mind that they are all referable entities.

Principle 3. *Embedded context.* A VSM-term represents not just a specific entity, but a specific entity in a specific *local context*. One should see each VSM-term as the sum of an entity, and the particular situation, state, or details specified by all terms connected to it. Each VSM-term thus carries a distinct and individual local context. For instance, in Fig. 1f the first VSM-term represents not just a particular man, but a man in a particular situation of wearing a suit, while eating etc. with etc.

This view is special yet essential to VSM. It enables the clear construction of VSM-sentences that mention a same entity in multiple situations, e.g. before vs. after some event (Fig. 1e, Fig. 3b). By using two VSM-terms instead of one, one can fully detail the entity in both situations separately, and prevent ambiguity about what detail belongs to what situation. One simply connects each detail (extra terms) to the VSM-term for the relevant situation. The second term can be labeled 'it', to reflect natural language. Both terms must be connected with a *coreference* to declare that they represent the same entity, not two distinct ones; and the context of the second term builds on the first.

Implications are: 1) A connector never attaches to a whole triple/unit, but always to a single term (which represents the unit's full meaning, from that term's perspective). 2) A connector does not carry meaning itself; it only operates on meanings carried by individual terms. 3) Connectors (excluding coreferences) must never form loops, as these signal conflated contexts.

These principles define a recipe for adding more context-details to every VSM-term, repeatedly and meaningfully. This is illustrated with many examples, including several with genuine scientific knowledge: Fig. 1f, 2a, Fig. 3a, online[31].

Other elements of VSM

Bidents, *list-connectors*, *VSM-templates*, and term types are shown in Fig. 1c, 1d, 2b, 2c, and elaborated in Supplementary Information.

Discussion

Our increasing dependence on computer-assisted knowledge discovery makes it imperative that our scientific knowledge is made computable. Simplifying the task of encoding complex ideas into computation-ready form is therefore crucial, and underpins an effective digital transformation.

VSM aspires to be a key enabling technology, as an intuitive, universal bridge between domain-expert and computer comprehension. For the end-user, the way knowledge is structured and displayed can make all the difference for accepting and using a new method. A VSM-sentence can often be understood at first glance, when using synonymous word forms that make it resemble natural language. It can be understood unambiguously, by reading terms while visually following the connectors that make syntax explicit. For the computer, only clear interpretability matters, achieved via IDs and connectors. VSM's three semantic principles enable all this.

The few connector types and principles make VSM a quick-to-learn method for both reading and composing computable knowledge in any domain. Fig. 2d highlights VSM's conceptual clarity, and Fig. 5 compares VSM with other methods. Early adopters reported that VSM makes biological information's inherent complexity much easier to handle. This also indicates VSM's utility for displaying knowledge currently encoded in more complex form.

Application of VSM in curation projects will likely start from templates (Fig. 2b), which are easy-to-use and readily extensible. Semi-automatic filling of templates or VSM-sentence construction based on text-mining are appealing topics for future research. Given collections of diverse VSM-sentences, learning algorithms may someday recognize patterns of equivalent or related conceptual structures[40], to support input, querying, and machine reasoning. VSM's understandable, context-rich conceptual structures may even facilitate the development of both explainable machine learning and robust machine reasoning.

Eventually, VSM could enable an interdisciplinary platform for crowdsourced creation of digital summaries of scientific papers, where each paper would be summarized on a wiki-like page, in a form both human- and computer-understood. While *structured digital abstracts* have long been an aspiration[12], VSM provides a usable and capable technology to that goal. It is a broad, walkable bridge between human and computer understanding.

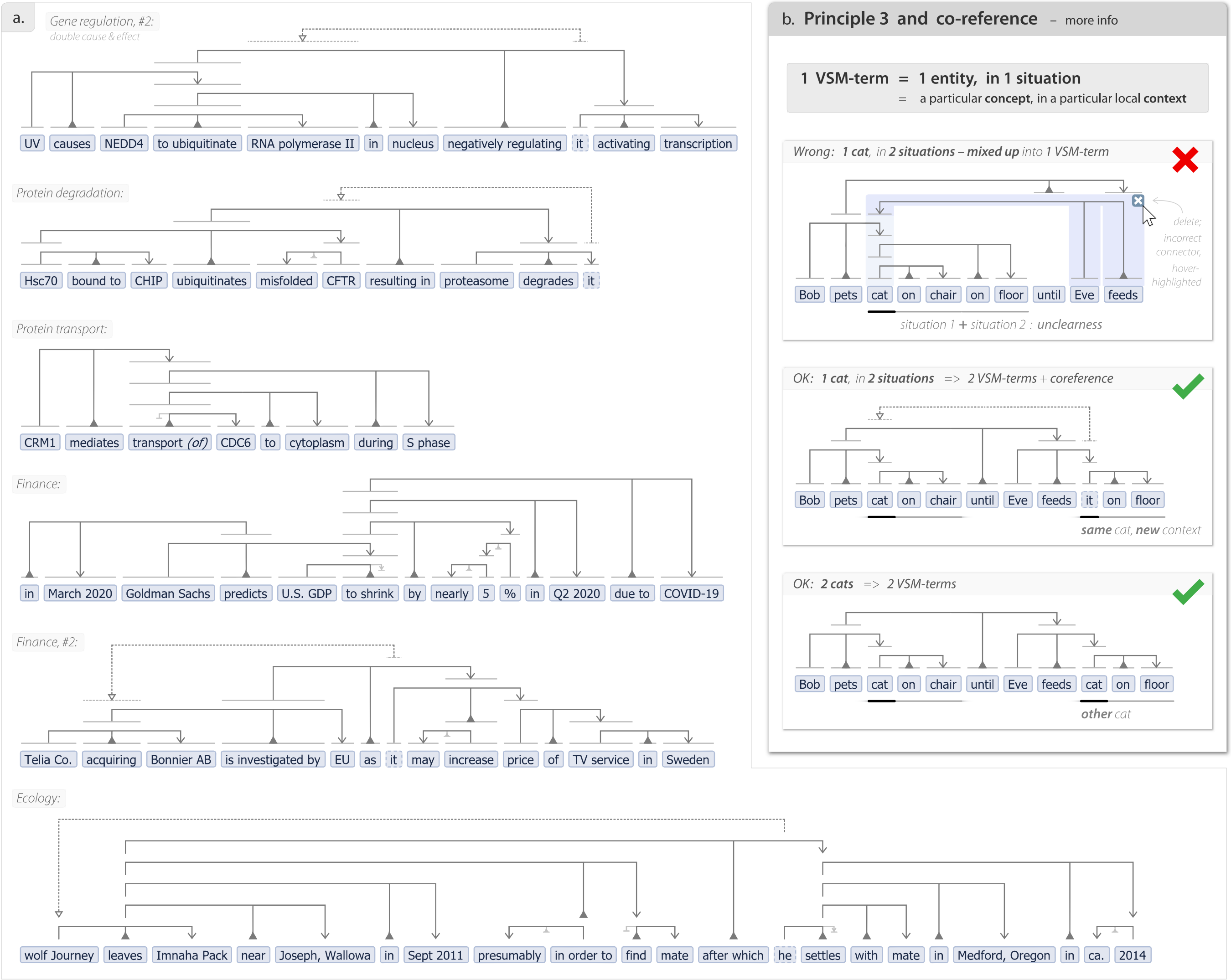
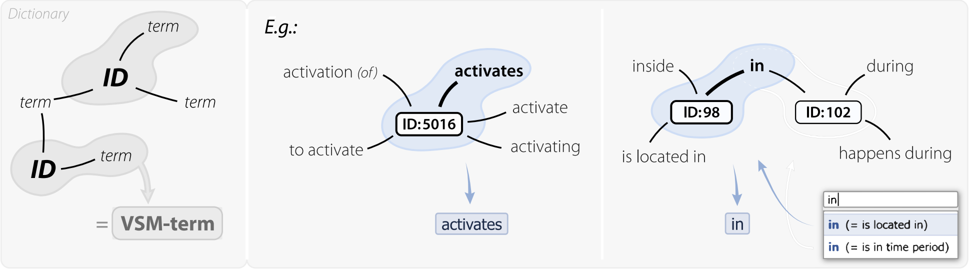


Fig. 3 | VSM's wide applicability, and notion of context. a, VSM-sentences can represent information on diverse topics and with any contextual depth. Some examples cover Life Science areas, where intricate systems of diverse components need to be modeled computationally, in order to understand or predict their behavior or response to changes. Other examples cover the financial world, where semantic web technologies are being applied to automate business processes and make real-time, well-informed decisions. – This illustrates that VSM can facilitate the digital transformation of real-world knowledge across domains. This may lead to the creation of larger volumes of such knowledge, and ultimately this may support, facilitate and stimulate the development of improved artificial reasoning over heterogenous knowledge. The recent outbreak of infectious disease may serve as an extra motivation to overcome the bottleneck of scarcity in human reasoning resources, w.r.t. utilizing all available knowledge to solve problems under time-sensitive circumstances. – A VSM-box container holding a VSM-sentence grows as needed, and automatically stacks connectors in natural-looking order for clarity. Stacking order does not affect meaning, only leg-to-term connections do. **b**, Full example on the notion of local context in VSM. Multiple connections to a term determine its contextualized meaning, all together, simultaneously. Therefore in the Wrong sentence (top of panel b), where both pets and feeds connect to a single cat term, this cat represents a cat in a situation of simultaneously being petted and being fed; i.e. its two contexts are conflated into one VSM-term. This makes it impossible to add further detail about the cat's state in either situation, without ambiguity. If one connects two locations to the single cat, it remains unclear when it is where. The first OK sentence solves this by using an extra VSM-term it, and connecting it to cat with a coreference. The resulting two VSM-terms represent the same entity, but in two different situations. This allows clear specification of the cat's location in each situation. The second OK sentence, with two cat VSM-terms and no coreference, describes two separate cats. This same method can be applied to e.g. statements about cause and effect, where an entity is described in the context of only the cause being applied, vs. later in the context of the effect happening; see also panel a. This applies also to non-temporal cases (e.g. in Ann thinks she is kind, the she represents a hypothetical version of Ann, not one later in time). Supplementary Information and Fig. S7 also explain how it (=child term) *inherits* context-details from cat (=parent term), in a unidirectional and overridable way, and how this viewpoint assists particular cases.

VSM is a language, communicated through a computer user-interface,
for representing any 'piece of information'
in a way that is easily understood by both humans and computers.

A **VSM-term** is a coupled **term + ID**, selected via a **user interface** :



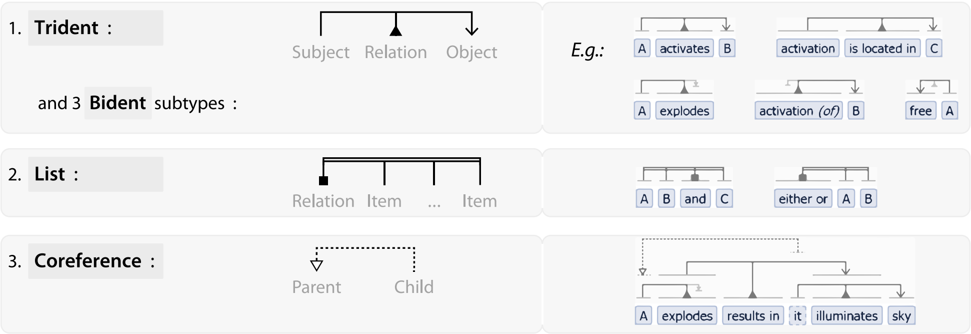
A **VSM-connector** is a visual element that **groups VSM-terms into subunits** that build up a VSM-sentence, and **assigns their role** in those units. Each connector **leg connects to a single term**. The **3 types** :

1. **Trident** :

and 3 **Bident** subtypes :

2. **List** :

3. **Coreference** :



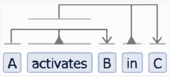
A **VSM-sentence** is built based on **3 Principles** :

PRINCIPLE 1: **Reformulate** the information; in a way that VSM-terms and -connectors can be used.

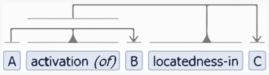
E.g.: from a **text**: *"We studied cell type C, and observed that molecule A had decisive role in activating protein B".*

to VSM

How a **human** *sees* it :



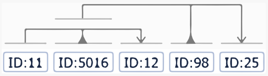
How humans & computers ***should think*** of it :



PRINCIPLE 2: Think of all **terms as nouns**, or 'things'.

- A term is only **interpreted** as a relation, under a relation-leg, **within** that connector's subunit.
- A term can **appear as** verb, adjective, etc, just to make a VSM-sentence more **human-readable**.

How a **computer** sees it :



How to *really think* of the connected terms :

1. *the A*, that activates B in C
2. *the activation*, of B by A, in C
3. *the B*, that is activated by A in C
4. *the being-located-in C*, of the activation of B by A
5. *the C*, in which A activates B

PRINCIPLE 3: Connectors make **terms more specific**.

- They are **operators** that **enrich all** terms' meaning, by placing them in each other's **context**.
- This forbids 'loops' among tri-/bident/list connectors.

By combining Principle 2 and 3,

One can always **meaningfully connect more terms to *any* term**, which makes each term represent a **more precisely specified** 'thing'.

← Fig. 4 | Overview of VSM-terms, VSM-connectors, and VSM Principles. – VSM-terms represent a term string and a coupled identifier (ID) (a unique code that represents one particular meaning). Term-strings and IDs are provided by dictionaries, and (especially with multiple dictionaries combined) may be in a many-to-many-relationship. – VSM-connectors assign VSM-terms to basic units. Each connector type reflects a distinct, basic conceptual unit-type that is used in VSM, while being intuitive in human understanding. They represent triples, pair types, lists, and linked concepts across contexts. We emphasize that this apparent grouping is not done in the sense, or with the purpose, of subsequently building groups of groups. Instead, each connector only assigns *relative roles* to individual VSM-terms, relative to each other. For example, `activates` is a Relation relative to `A` and `B`, while simultaneously, it is also a Subject relative to `in` and `C`. This enables the assembly of larger conceptual structures, not built by bagging terms into additional, referable group-entities that need management (like 'blank nodes' in RDF), but built via targeted connection to any VSM-term, no matter in what other groups it may already play a role. The semantics of this mechanism is governed by Principle 3, and is simple to follow for people with just this guideline: *think of each VSM-term as a specific thing, in its own situation or local context defined by all its connected terms*. In a sentence with five VSM-terms, this results in five specific concepts, each contextualized by the other four (context is also received from indirectly connected terms), and each representing the VSM-sentence's complete message from that term's own perspective. – The three VSM Principles (rephrased from the main text, yet expressing the same idea) define how VSM-terms and VSM-connectors work together semantically and guide the creation of clear VSM-sentences, ready for computation.

	Ease of use to build complex information	Method of semantic expression	Flexibility of structure, expressiveness	Lexical or connectivity representation	Has simple 'subject-relation-object' triples?	Method of composition to go past simple triples	Computer-understood	Easy for humans to understand meaning
VSM	High	Bottom-up	Infinite (by recursively adding connectors)	Graph + linear hybrid	Yes, plus binary and n-ary	Targeted attachment to any term in a triple or subunit, to enrich meaning	✓	✓
Controlled language	Low	Bottom-up	High (but many rules to consider)	Linear, disambiguated by rules and keywords/symbols	No	-	✓	✓
Spreadsheet or database tables	High	Bottom-up, semantics encoded in column headers	Low (if using 1 table) / medium (fixed after table design)	Table, forms	No, relations encoded in column headers or fields	-	✓ (good when using ontology terms)	✓
RDF	Low	Bottom-up	Infinite	Graph, through XML or controlled language like Turtle	Yes, plus a way to imitate n-ary	Various, e.g. reified triple as part of other triple	✓	✕ (less, the more reification)
Text annotation	-	-	-	Tags, can be graph	-	-	✓	✓
Parse tree (text mining)	-	Top-down, as result of parsing	-	Graph	No, encodes semantic roles in branch types	-	- (computer-generated)	-
Natural language	High	-	Infinite	Linear, ambiguous	Yes, but riddled with exceptions and figures of speech	-	✕	✓

Fig. 5 | Comparison of VSM with other knowledge representation methods. – VSM is a knowledge representation and entry method, most closely related to controlled languages, table-based entry methods, and RDF. – VSM differs from controlled languages: it replaces their many fixed rules by just a few connector types, and it replaces their fixed keywords and symbols by an approach that treats all VSM-terms equivalently (Principle 2). VSM supports synonymous word forms for readability, it places all expressed meaning into terms alone, and it uses elementary connector types that enable structural consistency (Supplementary Information Fig. S5b, S5c). – Table-based entry methods are easy to use but poor on inherent semantics. VSM-templates copy their ease-of-use, but provide seamless extensibility, immediate clarity of how terms relate, and ontology-ID lookup. – RDF (Resource Description Framework) achieves high expressivity through triples which are also present in VSM. But while RDF is designed for IT experts, VSM is designed with a focus on usability for other scientific domain-experts as well: the biologist, chemist, etc. curator. Next to triples, VSM includes other basic units for grouping terms, and adds a view of semantics where terms are embedded in an own, local context (Principle 3). Both are essential for representing larger conceptual ideas as structured units of information, with both ease and clarity. – VSM is much less related to natural language tagging or analysis tools, despite its ability for resemblance to natural language (Principle 1). In particular, VSM should not be confused with text annotation (the tagging of entities and relations in free-text sentences, as how they appear in a paper), and VSM should not be confused with parse trees (generated by text-mining algorithms after top-down syntactic analysis of natural language; in contrast to bottom-up semantic knowledge construction with VSM).

Acknowledgements We are indebted to Florian Leitner, Erick Antezana, Heri Ramampiaro, Rune Sætre, and Maria K. Andersen for valuable feedback on earlier versions of the text. We thank scientists at NTNU (Norwegian University of Science and Technology), EBI (European Bioinformatics Institute), ISB (International Society for Biocuration), GRECO (Gene Regulation Consortium), UC Berkeley / LNBL (University of California / Lawrence Berkeley National Laboratory), and many others for discussions. Among them, we thank especially Chris Mungall, Vasundra Touré, John Zobolas, Astrid Lægreid, Marcio L. Acencio, Sushil Tripathi, Liv Thommesen, Arne Jenssen, Ane M. Gabrielsen, Sophia Efstathiou, Paul D. Thomas, Livia Perfetto, and Ruth Lovering for in-depth feedback and/or encouragement.
Funding was provided intermittently by FUGE (Functional Genomics Res. Mid-Norway), NTNU, NFR (Res. Council of Norway) [247727/O70], COST (European Coop. in Science and Technology) [CA15205], Patreon.com/stevencruy, and Steven Vercruysse [2011-'16,'20].
Author contributions: S.V. conceived, designed, and developed VSM, and wrote the manuscript. M.K. guided and enabled the project, and co-wrote the manuscript.

Supplementary information is available for this paper at https://doi.org/10.20944/preprints202007.0486.v2.
Correspondence should be addressed to S.V. or M.K.

1. van Strien, N. M., Cappaert, N. L. & Witter, M. P. The anatomy of memory: an interactive overview of the parahippocampal-hippocampal network. *Nat. Rev. Neurosci.* **10**, 272–282 (2009).
2. Flobak, Å. et al. Discovery of Drug Synergies in Gastric Cancer Cells Predicted by Logical Modeling. *PLoS Comput. Biol.* **11**, e1004426 (2015).
3. Tripathi, S. et al. Gene regulation knowledge commons: community action takes care of DNA binding transcription factors. *Database (Oxford)* **2016**, baw088 (2016).
4. Villa, F., Athanasiadis, I. N. & Rizzoli, A. E. Modelling with knowledge: A review of emerging semantic approaches to environmental modelling. *Environmental Modelling & Software* **24**, 577–587 (2009).

5. Mungall, C. J. et al. The Monarch Initiative: an integrative data and analytic platform connecting phenotypes to genotypes across species. *Nucleic Acids Res.* **45**, D712–D722 (2017).
6. Antezana, E., Mironov, V. & Kuiper, M. The emergence of Semantic Systems Biology. *N Biotechnol* **30**, 286-290 (2013).
7. Menezes, T., & Roth, C. Semantic Hypergraphs. Preprint at https://arXiv.org/abs/1908.10784 (2019).
8. Schlegel, V., Lang, B. & Freitas, A. Vajra: Step-by-step Programming with Natural Language. *Proceedings of the 24th International Conference on Intelligent User Interfaces*, ACM (2019).
9. Browne, O., Krdzavac, N.B., O'Reilly, P., & Hutchinson, M. Semantic Ontologies and Financial Reporting: An Application of the FIBO. *JOWO* (2017).
10. International Society for Biocuration. Biocuration: Distilling data into knowledge. *PLoS Biol.* **16**, e2002846 (2018).
11. Krallinger, M. et al. The CHEMDNER corpus of chemicals and drugs and its annotation principles. *J Cheminform* **7**, S2 (2015).
12. Superti-Furga, G., Wieland, F. & Cesareni, G. Finally: The digital, democratic age of scientific abstracts. *FEBS Lett.* **582**, 1169 (2008).
13. Ashburner, M. et al. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat. Genet.* **25**, 25–29 (2000).
14. Huntley, R. P. & Lovering, R. C. Annotation Extensions. *Methods Mol. Biol.* **1446**, 233–243 (2017).
15. Kerrien, S. et al. The IntAct molecular interaction database in 2012. *Nucleic Acids Res.* **40**, D841–6 (2012).
16. Meldal, B. H. et al. The complex portal--an encyclopaedia of macromolecular complexes. *Nucleic Acids Res.* **43**, D479–84 (2015).
17. Bovolenta, L. A., Acencio, M. L. & Lemke, N. HTRIdb: an open-access database for experimentally verified human transcriptional regulation interactions. *BMC Genomics* **13**, 405 (2012).
18. Perfetto, L. et al. SIGNOR: a database of causal relationships between biological entities. *Nucleic Acids Res.* **44**, D548–54 (2016).
19. Touré, V. et al. The Minimum Information about a Molecular Interaction Causal Statement (MI2CAST), *Bioinformatics*, , btaa622, (2020).
20. Wilkinson, M. D. et al. The FAIR Guiding Principles for scientific data management and stewardship. *Sci Data* **3**, 160018 (2016).
21. Dutch Techcentre for Life Sciences. The FAIR Data Principles explained. https://www.dtls.nl/fair-data/fair-principles-explained (2017).
22. Mina, E. et al. Nanopublications for exposing experimental data in the life-sciences: a Huntington's Disease case study. *J Biomed Semantics* **6**, 5 (2015).
23. Clark, T., Ciccarese, P. N. & Goble, C. A. Micropublications: a semantic model for claims, evidence, arguments and annotations in biomedical communications. *J Biomed Semantics* **5**, 28 (2014).
24. Raciti, D., Yook, K., Harris, T. W., Schedl, T. & Sternberg, P. W. Micropublication: incentivizing community curation and placing unpublished data into the public domain. *Database (Oxford)* **2018** (2018).
25. Kuhn, T. & Dumontier, M. Genuine Semantic Publishing. *Data Sci.* **1**, 139–154 (2017).
26. Erhardt, R. A., Schneider, R. & Blaschke, C. Status of text-mining techniques applied to biomedical text. *Drug Discov. Today* **11**, 315–325 (2006).
27. Singhal, A. et al. Pressing needs of biomedical text mining in biocuration and beyond: opportunities and challenges. *Database (Oxford)* **2016** (2016).
28. Fuchs, N. E., Höfler, S., Kaljurand, K., Rinaldi, F., Schneider, G. (2005) Attempto Controlled English: a knowledge representation language readable by humans and machines. *Reasoning Web, Lecture Notes in Computer Science* **3564**, 213–250 (Springer, 2005).
29. Pratt, D., Biological Expression Language (BEL), https://bel.bio (2012).
30. Cyganiak, R. et al. Resource Description Framework (RDF), https://www.w3.org/TR/rdf11-concepts (2014).
31. Vercruysse, S. et al. The Vsmjs project. https://vsmjs.github.io (2020).
32. Vercruysse, S., Zobolas, J., Touré, V., Andersen, M. K. & Kuiper, M. VSM-box: General-purpose Interface for Biocuration and Knowledge Representation. Preprint at https://doi.org/10.20944/preprints202007.0557.v1 (2020).
33. Zobolas, J., Touré, V., Kuiper, M. & Vercruysse, S. UniBioDicts: Unified access to Biological Dictionaries. Preprint at https://doi.org/10.20944/preprints202007.0586.v1 (2020).
34. Touré, V., Zobolas, J., Kuiper, M. & Vercruysse, S. CausalBuilder: Bringing the MI2CAST Causal Interaction Annotation Standard to the Curator. Preprint at https://doi.org/10.20944/preprints202007.0622.v1 (2020).
35. Salvadores, M., Alexander, P. R., Musen, M. A. & Noy, N. F. BioPortal as a Dataset of Linked Biomedical Ontologies and Terminologies in RDF. *Semant Web* **4**, 277–284 (2013).
36. UniProt Consortium. UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Res.* **47**, D506–D515 (2019).
37. Brown, G. R. et al. Gene: a gene-centered information resource at NCBI. *Nucleic Acids Res.* **43**, D36–42 (2015).
38. Vercruysse, S. & Kuiper, M. Jointly creating digital abstracts: dealing with synonymy and polysemy. *BMC Res Notes* **5**, 601 (2012).
39. Tomlin, R. S. *Basic Word Order: Functional Principles.* (Croom Helm, 1986).
40. Agarwal, B., Ramampiaro, H., Langseth, H. & Ruocco, M. A deep network model for paraphrase detection in short text messages. *Information Processing & Management* **54**, 922–937 (2018).