Executive Summary

In this review, we bring together conceptual representations relevant to concept creation in the scope of ConCreTe. The conceptual representations reviewed are organized in accordance with two important perspectives on the distinctions between them. One distinction is between symbolic, spatial, and connectionist representations. The other is between descriptive and procedural representations. These two distinctions are orthogonal. Moreover, conceptual representations used in particular creative domains, i.e., language, music, image, and emotion, are reviewed separately. For each representation reviewed, we also cover the inference it affords, the computational means of building it, and its application in concept creation. In the end, we propose a high-level categorization of concept formation, and indicate directions of future research, as identified during this review, according to the proposed categories.
The ConCreTe Consortium has addressed all comments received, making changes as necessary. Changes to this document are detailed in the change log table below.

### Change log

<table>
<thead>
<tr>
<th>Date</th>
<th>Editor</th>
<th>Summary of changes made</th>
</tr>
</thead>
<tbody>
<tr>
<td>February 27, 2014</td>
<td>Ping Xiao</td>
<td>First proposal on the organization of conceptual representations</td>
</tr>
<tr>
<td>April 4, 2014</td>
<td>All partners</td>
<td>Contributions on various conceptual representations</td>
</tr>
<tr>
<td>April 29, 2014</td>
<td>Ping Xiao</td>
<td>A draft version, assembled partners' contributions, plus introduction</td>
</tr>
<tr>
<td>April 30, 2014</td>
<td>Pablo Gervás</td>
<td>Feedback to the draft</td>
</tr>
<tr>
<td>May 8, 2014</td>
<td>All partners</td>
<td>Revised the organization of conceptual representations</td>
</tr>
<tr>
<td>May 20, 2014</td>
<td>All partners</td>
<td>More contributions on various conceptual representations</td>
</tr>
<tr>
<td>June 2, 2014</td>
<td>Ping Xiao</td>
<td>A complete version, assembled and edited partners' contributions, plus executive summary, introduction, conclusions, and future work</td>
</tr>
<tr>
<td>June 5, 2014</td>
<td>Hannu Toivonen</td>
<td>Feedback and corrections for the complete version</td>
</tr>
<tr>
<td>June 12, 2014</td>
<td>All partners</td>
<td>Feedback and corrections</td>
</tr>
<tr>
<td>June 13, 2014</td>
<td>Ping Xiao</td>
<td>Complete version with all partners' inputs integrated</td>
</tr>
<tr>
<td>June 15, 2014</td>
<td>Nada Lavrač</td>
<td>Quality control</td>
</tr>
<tr>
<td>June 19, 2014</td>
<td>Hannu Toivonen</td>
<td>Conclusions and future work</td>
</tr>
<tr>
<td>June 20, 2014</td>
<td>Ping Xiao</td>
<td>Final version</td>
</tr>
<tr>
<td>June 26, 2014</td>
<td>Geraint Wiggins &amp; Sue White</td>
<td>Proof reading and correction</td>
</tr>
<tr>
<td>August 24, 2014</td>
<td>Ping Xiao</td>
<td>Figure permissions added</td>
</tr>
</tbody>
</table>
# Contents

1 Introduction 3

2 Symbolic Representations 5
   2.1 Association ................................. 5
   2.2 Semantic Relation ............................. 6
   2.3 Ontology .................................. 7
   2.4 Semantic Network ............................... 8
   2.5 Bisociation ................................ 9
   2.6 Information Network ............................ 11

3 Spatial Representations 11
   3.1 Conceptual Space ............................. 12
   3.2 Vector Space Model (VSM) .................... 13

4 Connectionist Representations 15
   4.1 Artificial Neural Network (ANN) ................ 15
   4.2 Neural Assembly ................................ 18
   4.3 Deep Representation ............................ 19

5 Procedural Representations 20

6 Applications 23
   6.1 Language .................................. 24
      6.1.1 Word .................................. 24
      6.1.2 Word Cluster and Fuzzy Synset ............ 25
      6.1.3 Syntactic Category ....................... 26
      6.1.4 Word Association Graph .................. 27
      6.1.5 Graph-based Representation of Document .... 28
      6.1.6 Plan Operator ............................ 30
      6.1.7 Narratological Category .................. 30
   6.2 Music ...................................... 31
      6.2.1 Charm: A General Representation of Music .... 32
      6.2.2 Music Representations in Evolutionary Computation (EC) .. 35
   6.3 Image ..................................... 35
   6.4 Emotion .................................... 37

7 Conclusions and Future Work 38

References 40
1 Introduction

Concept, in its most common meaning, refers to “an abstract idea”\(^1\). The ConCreTe (Concept Creation Technology) project is about generating new ideas. This sense of concept is used in Boden’s [23] pioneering study on creativity. On the other hand, a concept, in cognitive science, is regarded as a representation, which makes explicit certain information, albeit the existent or non-existent, physical or abstract, atomic or complex. The difference between these two senses of concept is that one refers to an idea itself whereas the other refers to a representation of the idea. Both senses of concept are used in this review. Most of the time, the textual context would indicate which sense is in use. Otherwise, we will note explicitly.

The above two-sides-of-the-same-coin distinction draws attention to the fact that the same idea can be, or often is, represented in more ways than one. Then, why is one representation chosen over another, or why does a representation come into existence in the first place? The answers to these questions deal with what we want to do and what we can do with a representation. A representation usually describes some but not all aspects of an entity, and embodies one kind of understanding of the entity. A representation normally supports a limited number of processing options, which are further constrained by the available scientific or technical methods at the time. People choose to use a representation depending on how easily it achieves their goals.

For studying cognitive functions of humans (and other animals) and artificial systems which are capable of human cognitive functions, Gärdenfors [70] presented the theory of conceptual spaces as a representational tool of concepts, and argued that this level of representation, termed conceptual level, is situated between the symbolic level and connectionist level. In this review, we call the conceptual level ‘spatial’, in order to avoid the confusion with ‘conceptual representations’.

At the symbolic level, information is represented by symbols. Rules are defined to manipulate symbols. Within a symbolic representation, meaning is internal to the representation itself; symbols have meaning only in terms of other symbols, and not in terms of any real world objects or phenomena they may represent. Symbolic representations are often associated with Good Old Fashioned AI (GOFAI), yet symbolic representation in itself does not entail classic GOFAI methodology. An underlying assumption of GOFAI research is that human thinking can be understood in terms of symbolic computation, in particular, computation based on formal principles of logic. However, symbolic systems have proved less successful in modelling aspects of human cognition beyond those closely related to logical thinking, such as perception.

At the spatial level, information is represented by points or regions in a conceptual space, which is built upon quality dimensions with defined geometrical, topological or ordinal properties. Similarity between concepts is represented in terms of the distance between points

\(^1\)http://www.oxforddictionaries.com/definition/english/concept?q=concept.
or regions in a multidimensional space. This formalism offers a parsimonious account of concept combination and acquisition, both of which are closely related to conceptual similarity. Besides, by defining dimensions in terms of perceptual qualities, spatial representations are grounded in our experience of the physical world, providing a semantics closely aligned with a human sense of meaning.

At the connectionist level, information is represented by the dynamics over densely connected networks of primitive units. A particular strength of connectionist networks is their ability to adapt their behaviour according to observed data. Nevertheless, since the learned behaviour is represented as weightings between units in the network, they offer limited explanatory insights into the process being modelled.

The three levels of representations outlined above differ in representational granularity, and each level has its own strengths and weaknesses in modelling cognitive functions. Gärdenfors [70] stressed that different representational formalisms should be seen as complementary, rather than competing, methodologies. As such, choices of representation should be made in accordance with scientific aims and in response to the challenges of the particular problem at hand. Furthermore, the spatial level can unify traditional symbolic and connectionist representations, providing a means to develop hybrid representations that combine the strengths of the various approaches, as investigated by Aisbett and Gibbon [4].

The above three levels of representations form one perspective on conceptual representations. There is another perspective, especially prominent in the computing community, i.e., the distinction between descriptive and procedural representations. As the name indicates, a descriptive representation describes the artifact being represented. The description may be low or high level, complete or partial. A procedural representation, on the other hand, specifies a procedure, e.g., a program, that once executed produces the artifact being represented. Like descriptive representations, the procedure may be low or high level, complete or partial. Sometimes, it is virtually impossible to come up with a descriptive representation of something, while its procedural representation is commonly used. Also, for everything we know how to produce, there exists at least one procedural representation, though it might not be the representation we usually use. Like the three levels of representations introduced above, descriptive and procedural representations have their own advantages and deficiencies. Which representation to use depends on how well it satisfies specific goals. The distinction between descriptive and procedural representations is orthogonal to the distinction between symbolic, spatial, and connectionist representations. At each of the three levels, both descriptive and procedural representations exist or can be constructed.

In this review, we bring together conceptual representations relevant to concept creation in the scope of ConCreTe. Some of them have general presence in computer science; some of them originated from computational creativity research; and some of them are especially created for concept creation. Some of them represent information that is relevant to a broad range of creative tasks, while some of them are unique for certain creative domains. Some of
them are atomic, forming the building blocks of some other more complex representations. For each representation reviewed, we also cover the inference it affords, the computational means of building it, and its application in concept creation.

The conceptual representations included in this review are organized according to the two perspectives introduced above. Most of the conceptual representations proposed in the literature are descriptive. Therefore, descriptive representations are reviewed first, classified using Gärdenfors’ three levels: symbolic representations (Section 2), spatial representations (Section 3), and connectionist representations (Section 4). Procedural representations, across the three levels, are reviewed in Section 5. Section 6 introduces conceptual representations used in four popular domains of investigation, i.e., language (Section 6.1), music (Section 6.2), image (Section 6.3), and emotion (Section 6.4), in the computational creativity community. Each subsection focuses on representing information particular to a domain, while the conceptual representations introduced in previous sections are more general, not limited to a specific domain. The information in the four domains may have representations in all the three levels and both descriptive and procedural representations. In Section 7, we give conclusions and propose future work.

2 Symbolic Representations

Symbols represent objects, properties of objects, relationships, ideas and so forth. Some symbols might be better discussed within domains. For instance, in Section 6, we will see symbols used in the domains of language, music, image, and emotion, such as word, music note, and pixel matrix. These are atomic representations. Examples of more complex symbolic representations are plan operator, SVG (Scalable Vector Graphics) file, and emotional category. In this section, we present symbolic representations that are applicable to many domains, including association, semantic relation, ontology, semantic network, bisociation, and information network.

2.1 Association

Association means “something linked in memory or imagination with a thing or person; the process of forming mental connections or bonds between sensations, ideas, or memories”\(^2\). The notion of association assumes a connection between concepts \(C_1\) and \(C_2\), but does not assume any specific conditions on \(C_1\) and \(C_2\). Also the nature of the connection in terms of meaning or specification of relation is not in primary focus.

Since concepts are commonly referred to by words, word associations have long been the interest of researchers. We will introduce them in Section 6.1.4, as part of the conceptual representations in the language domain.

2.2 Semantic Relation

Broadly speaking, semantic relations are relations between meanings, as well as between meanings and representations. The number of semantic relations is virtually unlimited. Some important semantic relations are *synonymy*, *homonymy*, *antonymy*, *association*, *causal relation*, *hyponymy-hypernymy*, *instance_of relation*, *locative relation*, *meronymy-holonymy* and *temporal relation*. Besides, there are domain-specific relations, such as the *person_afflicted* relation in the medicine domain [185] and the *ingredient_of* relation in the food domain [231].

Formally, a semantic relation can be represented by a *triplet* \((C_1, R, C_2)\) (Fig. 1), where \(C_1\) and \(C_2\) are concepts and \(R\) indicates a relation between them.

\[
C_1 \xrightarrow{R} C_2
\]

Figure 1: Triplet.

Electronic dictionaries have been the primary resources exploited for collecting semantic relations [6, 29]. The automatic extraction of semantic relations is usually done with text corpora. Most research of this line [35, 116, 139, 148, 187] is based on Hearst’s [90] method of discovering discriminating lexical-syntactic patterns. Starting with a set of seed relation instances of a certain type (e.g., hyponymy), this method identifies sequences of text that occur systematically between the related arguments. The patterns discovered can be used in the automatic extraction of new relations. This idea was combined with distributional measures to improve not only hypernymy extraction [16], but also other relation types [21, 31, 42]. In addition, machine learning techniques were applied to the extraction of semantic relations from text. [45, 53, 73, 202] are fully supervised approaches, while [89] is an unsupervised approach. Weakly supervised approaches were used in extracting semantic relations from the web [2], in order to learn a great amount of facts of various kinds [11, 32, 59].

There are many tools developed for extracting semantic relations. For example, GATE\(^3\) is an open-source free software. TextRunner\(^4\) is an Open Information Extraction (OIE) system, which goes through a text corpus and extracts a large set of triplets, without any human input [12]. ReVerb is a tool that extracts relations from English sentences [60] and was used to extract triplets from Wikipedia\(^5\) and Clueweb\(^6\). Both TextRunner and ReVerb do not have any predefined relations. Additionally, Rusu et al. [186] investigated how particular text parsers influence the results of triplet extraction algorithms.

---

\(^3\)https://gate.ac.uk.
2.3 Ontology

“In the context of computer and information sciences, an ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. The representational primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members)” [82].

We can distinguish between upper and domain ontologies. Upper ontologies describe general entities that are not lexicalized in text (e.g., concepts such as ‘entity’, ‘object’ or ‘situation’). They are usually manually constructed. Examples of ontologies containing upper level ontologies are SUMO [154] and Cyc [121]. OpenCyc\(^7\) is a public version of Cyc (which is proprietary). Domain ontologies, on the other hand, contain concepts that can be expressed directly in text.

From another perspective, we can distinguish between formal, terminological, and prototype-based ontologies [19, 204] (Fig. 2).

**Formal ontologies** are represented in logic, using axioms and definitions. Their advantage is the inference mechanism, i.e., the properties of entities can be derived (see, in Fig. 2, how one can derive that ‘chili con carne’ is ‘non-vegetarian’ food). Nevertheless, high effort of encoding is needed and there is danger of running into inconsistencies.

**Terminological ontologies/Taxonomy** have concept labels and are partially specified by subtype-supertype (is_a or part_of) relations, but lack other axiomatic grounding. A well known example is WordNet\(^8\).

**Prototype-based ontologies** represent categories (concepts) with typical instances, chosen according to some similarity metrics. The advantage of this type of ontology is that they are easy to construct and maintain using clustering techniques. However, they lack concept labels and cannot be used in certain type of applications. Another disadvantage is that they do not allow inference. A special case of prototype-based ontologies are topic ontologies [67], where document topics are organized hierarchically.

In taxonomy/ontology learning from text, there are two main approaches. The *distributional approach* is based on the hypothesis that similar words occur in similar contexts [86] and therefore should be near each other in a taxonomy/ontology. In general, classification is used for ontology population and enrichment, while clustering is used for building ontologies from scratch. The other approach uses *linguistic patterns* that reflect some specific relationships between words, e.g., Hearst’s [91] patterns for hypernymy relations. One can extract from text the hypernyms of a list of terms (using manually found or automatically learned patterns) and then connect the hyponym-hypernym pairs obtained using graph-based techniques [225].

Ontologies are used in many computational creativity tasks. The combinations of the-

\(^7\)http://www.cyc.com/platform/opencyc.
\(^8\)http://wordnet.princeton.edu.
matically different ontologies can be used for dealing with analogies (relating different symbols based on their similar axiomatisation), metaphors (blending symbols from one domain into another and impose the axiomatisation of the first on the second), pataphors (blending and extending two domains with each other), or conceptual blending (blending and combining two domains for the creation of new domains) [117].

2.4 Semantic Network

Semantic networks represent semantic relations between concepts. Figure 3 shows a small semantic network, which represents the sentences “The bottle contains wine. Wine is a beverage.” The concepts, i.e., ‘bottle’, ‘wine’ and ‘beverage’, are denoted by nodes, and the relations between them, i.e., ‘contains/contain’ and ‘is_a’, are represented by directed edges. The meaning of a concept is defined in terms of its connections with other nodes (concepts).

Semantic network is used to represent human long-term memory in the Spreading Activation model [7, 40], which seeks to explain how the mind processes related ideas, especially semantic or verbal concepts. The model assumes that activating one concept implies the spreading of activation to connected nodes, making those memory areas more available for further cognitive processing. The activation decays over time, and the further it spreads, the weaker it is, which is usually modelled using a decay factor. Spreading activation has been central in many cognitive models [182] and widely used in other domains such as information retrieval [46].

An example of available semantic networks is ConceptNet [125], a semantic network of commonsense knowledge. In ConceptNet, nodes are semi-structured English fragments, including noun phrases, verb phrases, adjective phrases, and prepositional phrases. Nodes are interrelated by one of the twenty four types of semantic relations, such as IsA, PartOf, UsedFor, MadeOf, Causes, HasProperty, DefinedAs, and ConceptuallyRelatedTo, represented by directed edges. The early versions of ConceptNet were built on the data of the Open Mind Common Sense Project9, which collects commonsense knowledge from volunteers on the web by asking them to fill the blanks in sentences [200]. ConceptNet 510, the current version, extends the previous versions with information automatically extracted from Wikipedia, Wiktionary11 and WordNet [62, 140]. The information stored in ConceptNet follows a loose schema in which relations with the same name actually can represent different aspects. Additionally, concepts are often expressed in terms of different atomic representations (fly vs. fly_animal).

Semantic networks and ontologies (Section 2.3) are not always clearly different, but one way of seeing a difference is to consider semantic networks as knowledge bases, while ontologies provide generic information about the semantic relations and possible inference mechanisms.

2.5 Bisociation

Mednick [136] defined creative thinking as the ability to generate new combinations of distant associative elements (e.g., words) and introduced the Associative Creativity Theory which
explains how thinking about concepts that are not strictly related to the elements under investigation inspires unexpected, useful connections between elements and thus considerably improves a creative process. Koestler [114] identified the associations spanning over two different contexts as an important element of creativity, and called them *bisociations*. According to Koestler, a bisociation is a result of the lateral processes of the mind, when making completely new associations between concepts from contexts (domains/categories/classes) that are usually considered separate. A schematic representation of bisociation is presented in Figure 4, where concepts $C_1$ and $C_2$, from two different contexts $D_1$ and $D_2$, respectively, are bisociated.

\[
\text{(from context/domain } D_1) \quad C_1 \leftrightarrow C_2 \quad \text{(from context/domain } D_2) \]

Figure 4: Bisociation.

A number of bisociation discovery methods are based on graph representations. An example is the FP7 project BISON (Bisociation Networks for Creative Information Discovery)\footnote{Bisociation Networks for Creative Information Discovery: http://www.BisoNet.eu.}, where researchers developed methods of finding cross-domain connections that are potentially new discoveries [56]. It has been shown by Don R. Swanson [215] that bisociation discovery can be tackled by literature mining methods. Many researchers followed Swanson’s idea of searching for the linking terms between two domains, a sort of bridge (Fig. 5). With an ever increasing amount of literature available on-line, literature mining methods offer a valuable tool for knowledge discovery, as they allow fragments of knowledge to be identified, connected and combined in a new way. This has been proved mostly in the biomedical domain. A good overview of the literature-based discovery approaches and challenges is available in [26].

\[
\text{(from context/domain } D_1) \quad C_1 \leftrightarrow B \leftrightarrow C_2 \quad \text{(from context/domain } D_2) \]

Figure 5: Bisociation using a bridging concept $B$.

In addition to bridging concepts, Berthold [17] presented other types of bisociations, i.e., bridging graphs and bridging by structural similarity. He pointed out that bridging concepts and bridging graphs require that the two domains have certain type of neighbourhood relation, while bridging by structural similarity allows matching on a more abstract level. Another interesting option is *bisociative triplets*, combining pairs (or chains) of triplets spanning over
different contexts.

2.6 Information Network

An information network consists of entities (e.g., web pages) that are in some way connected to other entities (one page may contain a link to another page). Sun and Han [212] defined information networks as directed graphs where both nodes and edges have types and the edge type uniquely characterizes the types of its adjacent nodes. When there are more than one type of node or edge in an information network, the network is called an heterogeneous information network; otherwise, it is a homogeneous information network.

A large amount of human knowledge can, in some way, be expressed by heterogeneous information networks. Bibliographic information networks [18, 211] are networks connecting the authors of scientific papers with their papers. Online social networks represent the communication in online social platforms, e.g., Twitter\(^{13}\) and Facebook\(^{14}\). Biological networks contain biological concepts and the relations between them.

In homogeneous information networks, new knowledge can be discovered by node/edge label propagation [242], link prediction [1, 13, 146, 147], community detection [158, 239], and node/edge ranking [46, 108, 113, 115, 150]. These methods can be applied to heterogeneous information networks by simply ignoring the heterogeneous information altogether. This does, however, decrease the amount of information used and can therefore decrease the performance of the algorithms [48]. Approaches that take into account the heterogeneous information are therefore preferable, such as network propositionalization [83], authority ranking [211, 214], ranking based clustering [211, 213, 214], classification through label propagation [106, 109, 211], ranking based classification [211], and multi-relational link prediction [48].

3 Spatial Representations

Compared with symbolic and connectionist representations, the importance of spatial representations was raised recently by Gärdenfors [70]. His theory of conceptual spaces aims at unifying ideas and evidences from several disciplines, including philosophy, computer science, psychology, linguistics and neuroscience. Since before Gärdenfors' proposal, in the computation community, a spatial representation called Vector Space Model (VSM) has been a popular tool for modelling many different domains and applications [55]. In this section, we introduce these two kinds of spatial representations and their relevance to concept creation.

\(^{13}\)https://twitter.com.


3.1 Conceptual Space

Gärdenfors [70] proposed a geometrical representation of concepts, called conceptual spaces. A conceptual space is formed by quality dimensions, which “correspond to the different ways stimuli are judged to be similar or different”. An archetypal example is a colour space with the dimensions hue, saturation (or chromaticism), and brightness. Each quality dimension has a particular geometrical structure. For example, hue is circular, whereas brightness and saturation correspond with finite linear scales (Fig. 6). It is important to note that the values on a dimension need not be numbers. Quality dimensions may be grouped into domains. A domain is a set of integral (as opposed to separable) dimensions, meaning that no dimension can take a value without every other dimension in the domain also taking a value. Therefore, hue, saturation, and brightness in the above colour model form a single domain. A conceptual space is simply “a collection of one or more domains” [70]. For example, a conceptual space of elementary coloured shapes could be defined as a space comprising the above domain of colour and a domain representing the perceptually salient features of a given set of shapes.

![Figure 6: The color space.](image)

A property corresponds to a region of a domain in a conceptual space. A concept, represented in terms of its properties, corresponds to a region in a space, normally including multiple domains. Property is a special case of concept. For instance, the concept ‘red’ is a region in the color space. It is also a property of anything which is red. An object is a point in a space. The spatial location of an object in a conceptual space allows the calculation of distance between objects, which gives rise to a natural way of representing similarities. The distance measure may be a true metric, or non-metric, such as a measure based on an ordinal relationship or the length of a path between vertices in a graph. When calculating distance, the salience (weight) of the dimensions is varied. It is the context where a concept is used that determines which dimensions are the most prominent, and hence, have bigger weights.

Such spatial representations naturally afford reasoning in terms of spatial regions. Boundaries between regions are fluid, an aspect of the representation that may be usefully exploited by creative systems searching for new interpretations of familiar concepts. Besides, conceptual spaces are particularly powerful in dealing with concept learning and concept combination. However, the conceptual spaces theory imposes some constraints on what kinds of subspaces

12
can be considered concepts, i.e., requiring them to be convex, which may compromise its applicability in general.

Two approaches to the mathematical formalisation of Gärdenfors’ theory of conceptual space appeared in the literature, both building on an initial formalisation by Aisbett and Gibbon [4]. One strand of research, based on fuzzy set theory, is presented in detail by Rickard et al. [176], drawing on their previous work [174, 175]. Another strand of research, employing vector spaces, is presented by Raubal [169], with subsequent related work by Schwering and Raubal [194, 195], and Raubal [170, 171].

3.2 Vector Space Model (VSM)

Vector Space Models (VSM) are a well-known example of computational models using spatial representations. VSM was invented by Salton when building the SMART information retrieval system [188]. VSM takes the terms in a document collection as dimensions. Every document is represented by a vector of terms, where the value of each element is the frequency of the corresponding term in the document. Each term in a document has a different level of importance, which can be represented by additional term weights in each document vector. A popular weighting schema is TF-IDF (Term Frequency - Inverse Document Frequency) [191], which is based on the idea that terms that appear in many documents are less important for a single document. Besides, long documents generally have more distinct terms and higher frequency of terms than short documents. Therefore, document vectors are usually normalized to unit length [190]. To see how similar two documents are, the most commonly used similarity measure is the cosine value of the angle between the document vectors [189]. Moreover, the same matrix can be used to calculate the similarity between two terms (i.e., a term is represented by a vector over the documents they appear in). VSM assumes pairwise orthogonality between term vectors (the columns), which generally does not hold due to correlation between terms. The Generalized Vector Space Model (GVSM) provides a solution for this problem [167, 237, 236].

Based on VSM, Latent Semantic Analysis (LSA) is a widely used similarity measure of words or text passages [118]. LSA applies singular value decomposition (SVD) to a word-context matrix, in order to create a new space with a drastically smaller number of dimensions, called latent semantic space, which helps discover high-order co-occurrence. Here, ‘context’ refers to a usually small window of words around a term. Before applying SVD, each cell entry of a word-context matrix is first transformed from frequency to its log value, and then all cell entries for a given word are divided by the entropy for that word. The number of dimensions chosen for the latent semantic space is critical for its performance, but there is no principled way of doing it. Moreover, the dimensions in the new space do not have obvious interpretations. LSA is not limited to words and their contexts. It can be generalized to unitary event types and the contexts in which instances of the event types appear.

Probabilistic Latent Semantic Analysis (PLSA) [98] is a probabilistic variant of LSA.
PLSA fits a statistical latent class model on a word-context matrix using Tempered Expectation Maximization (TEM), instead of applying SVD. This process generates a latent semantic space with much fewer dimensions, which are topics each described by a set of words with a varied degree of membership to the topic. The number of dimensions in the new space is determined based on statistical theory for model selection and complexity control. A shortcoming of PLSA is that it is not suitable for assigning probability to a previously unseen document, which is addressed by Latent Dirichlet allocation (LDA) [22]. Furthermore, word vectors can be learned by neural networks, which is good at capturing syntactic and semantic regularities and much more computationally efficient than LDA [137, 138].

Extending these VSMs to model the compositional meaning of phrases and sentences (rather than individual words) is the subject of much current research, with a range of methods including hierarchical compression using neural autoencoders (e.g., [203]) and categorical combination using tensor operations (e.g., [39]).

The starting matrices of the aforementioned VSMs are word-context matrices. Besides, pair-pattern matrices have been used, whose rows correspond to pairs of terms and columns correspond to the patterns in which the pairs occur. Pair-pattern matrices are used to measure the semantic similarity of patterns and of relations between word pairs [123, 218]. Matrices are second-order tensor. Higher-order tensors, such as a word-word-pattern tensor [220], were also found useful in measuring similarity of words.

In the Computational Creativity community, Xiao and Blat [238] built a LSA space from Wikipedia to generate pictorial metaphors. Veale [224] constructed topic vectors for conceptual blending. Jurčič et al. [18] used a document-term matrix and centroid vectors to find bridging terms of two literatures. Venour, Ritchie, and Mellish [227] constructed a novel semantic space in which the distance between words reflects the difference in their styles or tones, in order to produce incongruity. Strapparava, Valitutti, and Stock [208] used LSA to compute lexical affective semantic similarity for generating advertising messages and their animations. Wong and Chun [235] used VSM to find semantically related sentences to generate ‘modern haiku’ poetry. Besides constructing VSMs from text, Thorogood and Pasquier [216] used vectors of low-level audio features to generate audio metaphors. De Melo and Gratch [49] used RGB vectors to evolve the emotional expressions of virtual humans. Maher, Brady and Fisher [130] used vectors of attributes (e.g., display area, amount of memory, and CPU speed) to measure surprise. Vector-space (geometric) representations of words have shown success at representing similarities and differences in meaning between lexical concepts (e.g., [141, 221]) and the regularities associated with these [138, 230]); this suggests they may be useful for application in computational creativity tasks [135].
4 Connectionist Representations

Connectionist representations are composed of interconnected simple units, featuring parallel distributed processing. Connectionist representations are at a lower level of abstraction than spatial representations. The most commonly used family of connectionist models are Artificial Neural Networks (ANNs). Besides, Donald Hebb [92] proposed that concepts are represented in brain in terms of neural assemblies. A more complex version of ANN, deep representation, provides representations at a series of abstraction levels.

4.1 Artificial Neural Network (ANN)

Although ANNs draw inspiration from biological human neurons, their usage is usually not intended to approach the complexity of the human brain. However, there are two major similarities between artificial and biological neural networks: first, the building blocks of both networks are simple computational units that are highly interconnected; and second, the connection among the units determines the function of the network.

ANNs can be viewed as weighted directed graphs in which artificial neurons are nodes and directed edges (with weights) are connections between neuron outputs and neuron inputs. Figure 7 shows several ANNs that have been developed through time. Based on the connection patterns (architecture), ANNs can be grouped into several categories: feed-forward neural networks, in which graphs have no loops; and recurrent (or feedback) neural networks, in which loops occur due to feedback connections.

In the most common family of feed-forward neural networks, called multilayer perceptron, neurons are organized into layers (input, hidden, and output layers) that have unidirectional connections between them. Different connectivity structures yield different network behaviors. Generally speaking, feed-forward networks are static, that is, they produce only one set of output values, rather than a sequence of values, for a given input. They are also memory-less in the sense that their responses to inputs are independent of the previous network states. Feed-forward neural networks are usually used as classifiers, by learning nonlinear relationships between inputs and outputs. In the context of computational creativity, a feed-forward neural network can be used as an evaluation mechanism of newly created concepts, having learned the characteristics of “good” and “bad” concepts beforehand.

Recurrent, or feedback, neural networks, on the other hand, are dynamic systems [210]. When a new input pattern is presented, the neuron outputs are computed. Because of the feedback paths, the inputs to neurons are then modified, which leads the network to enter a new state.

Hopfield Networks are recurrent neural networks, invented by John Hopfield in 1982 [100]. Connections between nodes are symmetrical and a node is not allowed to connect to itself. A Hopfield net of four nodes is shown in Figure 8. One of the most interesting properties of
Hopfield nets is their ability to store and retrieve patterns, working as an associative memory. Hopfield nets use an energy function to determine the current state of the network, and a learned pattern is remembered by descending a gradient of energy toward a local minimum, which is usually achieved using Hebbian learning [92].

The Boltzmann Machine [93] is another form of recurrent neural network, and can be seen as a stochastic and generative counterpart of Hopfield nets. The machine is composed of primitive computing elements called units that are connected to each other by bidirectional links. Figure 9 presents a Boltzmann machine with a few weight labels. A unit is always in one of two states, on or off. It adopts these states according to a probabilistic function of the states of its neighboring units and the weights of its links to them. A unit being on or off is
taken to mean that the system currently accepts or rejects some elemental hypothesis about the domain. The weight of a link indicates a pairwise constraint between two hypotheses. A positive weight indicates that the two hypotheses tend to support each other; if one is accepted, accepting the other should be more likely. Conversely, a negative weight suggests that the two hypotheses should not both be accepted.

The units in a Boltzmann Machine are divided into visible and hidden units. For the example in Figure 9, there are four visible units (white) and three hidden units (blue). Training a Boltzmann machine involves two iterating phases. One is the positive phase, when the visible units are given as input a binary state vector sampled from the training set. The other is the negative phase, when the network is allowed to run freely, i.e., the state of each unit is not determined by external data.

Although learning is impractical in general Boltzmann machines, it can be made quite efficient in Restricted Boltzmann Machine (RBM) [93], which does not allow intralayer connections between hidden units. After a RBM is trained, the activities of its hidden units can be treated as input data for training a higher-level RBM. This method of stacking RBMs makes it possible to train many layers of hidden units efficiently and is one of the most common strategies of deep learning. As each new layer is added, the overall generative model gets better.

Figure 9: A Boltzmann Machine.

Instead of following a set of rules specified by human experts, ANNs are able to learn underlying rules (e.g., input-output relationships) from a given set of representative examples. ANNs learn by iteratively updating the connection weights in the network, toward performing better in a specific task. Learning paradigm deals with what information is available to a network. There are mainly three learning paradigms: supervised, unsupervised, and hybrid. In supervised learning, there exist input-output pairs that can be used for training. In unsupervised learning there is no output available, and in hybrid learning there are some input-output pairs available for training. Learning rules specify how connection weights are updated. There are two basic types of learning rules: error-driven and Hebbian.

Error-driven learning In supervised learning, the network is given a desired output for each input. During the learning process, the actual output \( y \) generated by the network may
not equal the desired output \( d \). The basic principle of error-driven learning is modifying the connection weights to gradually reduce the error \((d - y)\). Perceptron and backpropagation \cite{210} are based on error-correction.

**Hebbian learning**

Hebb's \cite{92} postulate of learning is the oldest learning rule. Hebbian learning is based upon the following observation in neurobiological experiments: if the neurons on both sides of a synapse are activated synchronously and repeatedly, the synapse’s strength is increased. An important property of this rule is that learning is done locally, that is, the change in synapse weight depends only on the activities of the two neurons connected to it. Hopfield network \cite{92} is an example of using Hebbian learning.

Since ANNs can learn patterns of neuron activation (both simultaneous and sequential activations), they can be used to simulate creative processes, e.g., via combining two or more patterns into a single one, or creating a random variation of a learned pattern.

### 4.2 Neural Assembly

Donald Hebb \cite{92} proposed that concepts are represented in the brain in terms of neural assemblies. These assemblies arise over time, based on learning processes such as Hebbian learning. A neural assembly is a group of neurons that are strongly interconnected. As a consequence, when a part of an assembly is activated, e.g., by perception, it can reactivate the other parts of the assembly as well. An assembly arises when neurons are conjointly active during a particular process. For example, when we see an animal, certain neurons in the visual cortex will be active. But when the animal makes a sound, certain neurons in the auditory cortex will be active as well. Based on the process of Hebbian learning, the active neurons will become (more) interconnected. Over time, other neurons could become a part of the assembly as well, in particular when they are consistently active together with the assembly. Examples are the neurons that represent the word we use to name the animal, or neurons involved in our actions or emotions when we encounter it.

Figure 10 illustrates (parts of) a neural assembly that could represent the logo for the ConCreTe project. It would consist of the neurons involved in perceiving the logo, but also of neurons representing the words ‘concrete’ or ‘egg’, and neurons that represent our experience of the material concrete, e.g., the fact that it is ‘hard’.

The assembly for the ConCreTe logo illustrates an interesting aspect of conceptual representation with neural assemblies. The first concepts we develop will consist of representations of experiences related to the objects we encounter and the actions we engage in. But based on this initial set of concepts, we could develop conceptual representations that consist of a mixture of experiences and relations to other already familiar concepts. The neural assembly for the ConCreTe logo illustrates such a mixture. We recognize the logo because we are familiar with the shape of an egg and the nature of concrete as a substance (e.g., that it is hard). We are also familiar with the words ‘concrete’ and ‘egg’. In a rapid way, we can integrate these
different conceptual representations (assemblies) in a neural assembly for the ConCreTe logo.

[223] presented a neural architecture in which relations between conceptual representations based on neural assemblies could be formed. A fundamental characteristic of this architecture is that it forms (temporal or more permanent) interconnections between the neural assemblies representing the concepts. In this way, the connection structure of each of the assemblies remains intact, so that the experiences that gave rise to the conceptual representations are not lost.

The figure illustrates the relation ‘ConCreTe is nice’. In short, the neural assemblies representing the concepts ‘ConCreTe’, ‘is’, and ‘nice’ are interconnected in a neural blackboard. The blackboard consists of neural populations that represent the specific types of concepts and the relations between them. Thus, $N$ represents a noun, $V$ a verb, $S$ a clause and $Adj$ an adjective. The connections in the blackboard are gated connections (consisting of neural circuits). In this way, a temporal connection can be formed between the conceptual assemblies and populations of the same type in the blackboard (‘ConCreTe’ with $N$, ‘is’ with $V$, ‘nice’ with $Adj$) and between the type populations themselves, which results in the representation for the clause (relation) ‘ConCreTe is nice’.

In the ConCreTe project, we will develop a ‘sequential neural blackboard’ that allows the dynamic processing and generation of concept sequences.

4.3 Deep Representation

Deep representation is a recent family of connectionist representations, which attempts to model high-level abstractions of data using deep architectures [15]. Deep architectures are composed of multiple levels of nonlinear operations, such as neural nets with many hidden
layers, or complicated propositional formulae reusing many subformulae.

The human brain is organized in a deep architecture [197]. An input percept is represented at multiple levels of abstraction, each level corresponding to a different area of cortex. The brain also appears to process information through multiple stages of transformation. This is particularly evident in the primate visual system [197], with its sequence of processing stages, from detection of edges, primitive shapes, to gradually more complex shapes.

Deep representations are built with deep learning techniques [14]. The focus of deep learning is to automatically discover multi-level abstractions, from the lowest level features to the highest level concepts. Many deep learning algorithms are framed as unsupervised learning problems. Deep belief network [94, 95] is an example of a deep structure that can be trained in an unsupervised manner. A fast greedy algorithm for directed belief networks is proposed in [95], showing its equivalence with Restricted Boltzmann Machines (RBMs) (see, Section 4.1) or infinite directed networks with tied weights. Deep belief networks are based upon probabilistic approaches, whereas other approaches exists, such as auto-encoders which is based upon reconstruction-based algorithms, and manifold-learning which has roots in geometrical approaches.

Deep learning algorithms typically operate on multi-layer networks with fixed topology. Although the stacked layers may allow the network to effectively learn the intricacies of the input, the fact that it has a fixed topology imposes representation and learning limits a priori. In contrast, a deep learning algorithm for dynamic topologies, allowing the creation of new nodes or layers of nodes, would enable the creation of new concepts and new dimensions in a conceptual space.

5 Procedural Representations

A procedural representation specifies a procedure, e.g., a program, that once executed produces the artifact being represented. To illustrate the difference between descriptive and procedural representations, we will resort to an example in the musical domain, the task of evolving a sequence of pitches. Using a descriptive approach one could, for instance, use a vector of pitch values to represent this sequence. A procedural representation would, for instance, use a program to generate the sequence of pitches by performing a set of operations on sub-sequences. In Figure 11 we present an example of such an approach, the GP-Music system. “The item returned by the program tree in the GP-Music System is not a simple value but a note sequence. Each node in the tree propagates up a musical note string, which is then modified by the next higher node. In this way a complete sequence of notes is built up, and the final string is returned by the root node. Note also that there is no input to the program; the tree itself specifies a complete musical sequence” [110].

A descriptive representation is straightforward to conceive, therefore one should ponder about the motivation for using procedural representations. A program can take advantage of
Figure 11: Genotype of an individual in the GP-Music system, indicating how a sequence is built up [110]. Reproduced by permission.

the structure of an artifact - e.g., repetition of note sequences, relations between sequences, cycles, etc. - and as such the size of the procedural representation of an artifact that has structure tends to be significantly smaller than the size of its descriptive representation. Additionally, it is also easier to induce structural changes, in case of creating new concepts.

Procedural representations are particularly popular for image generation in Evolutionary
Computation (EC). Many of them are expression-based. Some notable examples are created by Sims [199], Rooke [184], Machado and Cardoso [127], Hart [88], Unemi [222], and Saunders and Gero [192]. In Figure 12 we present examples of symbolic expressions and corresponding images. Machado and Nunes [128] evolved non-deterministic context free grammars. The grammars are represented by means of a hierarchic graph, which is manipulated by graph-based crossover and mutation operators. One of the novel aspects of this approach is that each grammar has the potential to represent, and generate, a family of akin shapes (Fig. 13). Byrne et al. [28] evolved architectural models using grammatical evolution. Grammatical evolution is a grammar based form of Genetic Programming (GP), replacing the parse-tree based structure of GP with a linear genome. It generates programs by evolving a integer string to select rules from a user-defined grammar. The rule selections build a derivation tree that represents a program. Any mutation or crossover operators are applied to the linear genome instead of the tree itself. McDermott [134] also used grammatical evolution to evolve graph grammars in the context of evolutionary 3D design. In this case the authors evolved shape grammars [206], which is one of the distinctive aspects of the work. Greenfield [80] used GP to evolve controllers for drawing robots. To evolve the controllers he resorted to an assembly language where each statement is represented as a triple. The programs assume the form of a tree.

Figure 12: Examples of expression-based genotypes and corresponding phenotypes [129]. Reproduced by permission.

One of the first, if not the first, evolutionary approaches to music composition resorts to a procedural representation. Horner and Goldberg [103] used Genetic Algorithm (GA) for
Figure 13: Examples of non-deterministic grammar, their corresponding tree-like shapes, and their graph representation [128]. Reproduced by permission.

thematic bridging evolving sequences of operations that transform an initial note-sequence into a final desired sequence within a certain number of steps. Putnam [166] was one of the first to use GP for music generation purposes. He used the traditional GP tree-structures to interactively evolve sounds. Spector and Alpern [205] used GP to evolve programs that transform an input melody by applying several transformation functions (e.g., invert, augment, transpose). The work of Johanson and Poli [110] constitutes another early application of GP in the music domain. Hornel and Ragg [101] evolved the weights of neural networks that perform harmonization of melodies in different musical styles. McCormack [132] explored stochastic methods for music composition and proposed evolving the transition probability matrix for Markov models. Monmarché et al. [143] used artificial ants to build a melody according to transition probabilities while also taking advantage of the collective behavior of the ants marking paths with pheromones. They evolved graph like structures, the vertices are Midi events and a melody corresponds to a path through several vertices (Fig. 14). McCormack [132] focused on grammar based approaches for music composition, exploring the use of L-systems. In a previous work [131], McCormack used L-systems for evolving 3D shapes.

6 Applications

In the previous sections, we introduced conceptual representations at each of the symbolic, spatial and connectionist levels, as well as both descriptive and procedural representations. The representations reviewed so far are domain-generic. Nonetheless, information, or knowledge, is also commonly organized in terms of domains. In this section, we review conceptual representations used in four popular domains of investigation, i.e., language, music, image, and emotion, in the computational creativity community. The information in these four domains may have representations at all the three levels and both descriptive and procedural representations.
6.1 Language

In the language domain, one of the atomic conceptual representations is word. A single word is ambiguous. Word cluster and fuzzy synset are representations aiming at expressing meanings precisely. Syntactic categories are labels which denote a group of words and/or phrases that share common characteristics. More complex conceptual representations are built upon words, such as word association graphs and graph-based representations of documents. The above are conceptual representations used in a broad range of language-related tasks. Besides, we review two representations used in narrative generation, plan operator and narratological category.

6.1.1 Word

Natural language is typically used to refer to a concept, which can be denoted by a word. Traditionally, words are collected in dictionaries. Broad-coverage lexical knowledge bases (LKBs) are computational resources.

WordNet [62, 140] is a manually constructed LKB, designed with the inspiration of current psycholinguistic theories of human lexical memory. The most ambitious feature of WordNet is its attempt to organize lexical information in terms of word meanings, rather than word forms. English nouns, verbs, adjectives, and adverbs are organized into synonym sets (synsets), each representing one underlying lexical concept (word sense). Another prominent feature of WordNet is that synsets are linked by conceptual-semantic and lexical relations, such as synonymy, antonymy, hypernymy, hyponymy, holonymy, meronymy, attribute, cause, and domain. WordNet is by far the most popular among all resources that are indexed by concepts. Given its success in English, the WordNet model has been adapted for many other
languages [24].

To bridge the gap between linguistic and world knowledge, WordNet has been linked to or integrated with other knowledge bases, such as other lexical-semantic resources (e.g., FrameNet [10], VerbNet [198], and Wiktionary [84]), an online encyclopedia Wikipedia, an upper ontology SUMO [153], and a descriptive ontology DOLCE [69]. BabelNet [145] and YAGO [97, 209] are two of such efforts.

As the manual creation of knowledge bases is a time-consuming and tedious task, the developers of several LKBs, including wordnets, turn to Information Extraction (IE) techniques for text, in order to create new resources or enrich the existing ones. Typical IE systems acquire concept instances and information about them from large collections of text. They typically represent concepts as terms, which are lexical items identified by their orthographic form. Semantic relations are denoted by relational triplets (see, Section 2.2). The problem is that a simple term is usually not enough to unambiguously refer to a concept, because the same word might have different meanings and different words might have the same meaning. Pantel [151] introduced the task of ontologising, which aims to associate terms, extracted from text, to their meanings, represented, for instance, as a synset in a wordnet. Pantel and Pennacchiotti [157] presented two methods that take advantage of the structure of WordNet to ontologise relational triplets. The anchor approach assumes that terms related in the same way to a fixed term are more plausible to describe the same sense. Therefore, to select a suitable synset for a term, it exploits extracted triplets of the same type sharing one term argument. The clustering approach selects suitable synsets using generalisation through hypernymy links in WordNet. Alternative algorithms for ontologising semantic relations were presented in [76]. They do not consider the extraction context as cues for disambiguation and do not use the WordNet structure directly. Instead, they exploit all the extracted information for computing similarities between the term arguments and the terms in candidate synsets.

Knowledge bases structured in words are resources overly used in text-based creative systems, including poetry generation [3, 41, 74], narrative [71, 177], computational humor [179], etc.

6.1.2 Word Cluster and Fuzzy Synset

As natural language is ambiguous, in opposition to formal languages, one word is often not enough to refer to a specific concept. Whether with explicit or implicit relations, a group of words is a common alternative to describe a concept.

A notable resource of word clusters is Roget’s Thesaurus. The original edition was published in 1852 [183] and each revision has been larger. It has a hierarchical classification structure, with several primary classes and each class has sections, subsections and heads. Semantically related words and phrases are organized in groups led by head words. The 1911
Most of the computational work on harvesting word clusters relies on Harris distribu-
tional hypothesis [87], which assumes that similar words tend to occur in similar contexts.
After defining the context of a word, these works generally follow a procedure to cluster
words according to their distributional similarity. Earlier approaches for clustering similar
nouns [178, 180] were weakly supervised. They used bootstrapping algorithms that started
with a set of seed words belonging to the same category (e.g., ‘airplane’, ‘car’, ‘jeep’, ‘plane’,
and ‘truck’), in order to discover more members of this category. Lin [122] proposed a fully
unsupervised clustering approach. In his work, each word is represented by a vector with
the contexts where it occurs (a VSM, see, Section 3.2), and similarities are computed with
additional information about syntactic dependencies. Clustering by Committee (CBC) [124]
is another unsupervised method.

A special case of word cluster is synset. New synonyms can be discovered from raw text
using semantic relatedness measures (see, Section 6.1.4). Apart from raw text, synonyms can
be extracted from dictionaries [77], especially from definitions having only one word or using
the “same as” pattern.

From a linguistic point of view, word senses are not discrete and cannot be separated
with clear boundaries [96, 112]. Sense division in dictionaries and ontologies is thus often
artificial. This also applies to concepts, and representing them as crisp objects does not
reflect the way humans organize knowledge. A more realistic approach would be adopting
models of uncertainty, such as fuzzy logic. [226] represents word sense classes as fuzzy clusters,
where each word has an associated membership degree. Other works represent concepts
as fuzzy synsets. The fuzzy membership of a word in a synset can be interpreted as the
confidence level about using this word to indicate the meaning of the synset. For instance,
Swesaurus [25], the Swedish wordnet, integrates the notion of fuzzy synsets by including
information from Synlex16, a lexical resource created by asking users of a Swedish-English
dictionary to judge the degree of synonymy of random, automatically generated synonym
pairs. Furthermore, fuzzy synsets were discovered automatically from synonym networks,
extracted from Portuguese dictionaries [75]. Based on their adjacencies, the applied algorithm
computes the similarity of each word in the network with all of the other words. It then
exploits the similarity matrix to discover synonym clusters (synsets). Finally, the similarity
values with each other word in a cluster were used as the fuzzy membership.

6.1.3 Syntactic Category

A syntactic category is a set of words and/or phrases in a language which share a significant
number of common characteristics [36]. In this sense, they are interesting abstractions to be

15http://www.gutenberg.org/ebooks/10681.
considered for concept creation.

In the field of Natural Language Processing (NLP), syntactic categories are represented as syntactic rules, which associate a particular syntactic label with a set of possible syntactic constructions. A natural language parser based on constituent grammar is a program that works out the grammatical structure of sentences in terms of a syntax tree. Such a tree is built from syntactic rules as described above. Probabilistic parsers use knowledge of language gained from hand-parsed sentences to try to produce the most likely analysis of new sentences. These statistical parsers still make some mistakes, but commonly work rather well. Their development was one of the biggest breakthroughs in NLP in the 1990s. Existing efforts at automating syntactic analysis of text rely on human annotation to provide the set of syntactic categories, both at the terminal and non-terminal level. Technologies capable of creating the concepts needed to represent these terminal and non-terminal categories would be very useful.

6.1.4 Word Association Graph

Word association is a pair of words that are related in some way. Word associations have been collected in psychological experiments where a word (stimulus) is presented to people who are asked to say or write down the word which first comes to their minds. The frequencies of each distinct response are counted. This experiment has been carried out with numerous people using different stimuli. Considerable agreement among people was found as to the most popular responses. The responses are in various relations with the stimulus, such as synonymy, antonymy, meronymy, hierarchic relations (category-exemplar, exemplar-category and category coordinates), idiomatic, and functional relations [47]. Word associations reflect to a large degree the semantic structure of the lexicon, but also the dynamics of word activation and retrieval (e.g., word class, word frequency, semantic density, etc.) [164]. There are two large collections of word associations. One is the Edinburgh Associative Thesaurus (EAT)\textsuperscript{17}. In its construction, 8,400 stimulus words were used. The experiment subjects were primarily undergraduates from various British universities. The other collection is the University of South Florida Free Association Norms\textsuperscript{18}, which is the largest database of free association ever collected in the United States. Total 5,019 stimuli were used. For instance, the most frequent responses for the stimulus ‘apple’ are ‘pie’, ‘pear’, ‘orange’, ‘tree’, ‘core’, ‘fruit’, ‘juice’, ‘eat’, ‘eye’, and ‘tart’.

Words can be connected according to their associations, which becomes a graph. In this graph, a word, or the concept it denotes, is defined by the connections it has with other words, i.e., ‘car’ is defined by its associations to ‘drive’, ‘road’, ‘vehicle’, ‘traffic’, ‘personal’, ‘tires’, ‘driver’, etc.

The computational work on harvesting word associations is closely related to calculating semantic relatedness. In general, semantic relatedness is measured by using distance measures

\footnote{\url{http://www.eat.rl.ac.uk}.}
\footnote{\url{http://w3.usf.edu/FreeAssociation/}.}
in certain materialized conceptual space, mainly knowledge bases (KBs) and raw text. KBs include dictionary, thesaurus, and ontologies, which are represented as graphs or networks. Hence, the semantic relatedness measures using KBs are path related calculation. With raw text, there are distributional measures based on the frequency of word co-occurrence, such as Latent Semantic Analysis (LSA) [50], Pointwise Mutual Information and Information Retrieval (PMI-IR) [219], and Normalized Google Distance (NGD) [37]. Log-likelihood ratio is a non-parametric statistical test for co-occurrence analysis. Using log-likelihood ratio for word co-occurrence analysis was proposed by Dunning [57] who showed, in particular, that log-likelihood ratio does not overestimate the importance of very frequent words like some other measures. Evaluated against human coded golden standards, the state-of-the-art semantic relatedness measures achieve about 80% accuracy [27, 68, 78, 172, 241]. The size and content (single or multiple topical) of the text corpora affect the accuracy of the semantic relatedness values calculated. Usually, larger corpora are preferred, since they cover more human experience.

Word associations have wide usage in NLP tasks. Roark and Charniak [181] used co-occurrence statistics for detecting potential entries for a word category. Word association graphs are often used together with other lexical resources. For instance,Navigli [144] used a semantic network-based approach for word sense disambiguation, where different machine-readable dictionaries and WordNet were translated into a graph. Then the word sense disambiguation task was solved by analysing the cycles and quasi-cycles in the corresponding graph. In the computational creativity area, word association graphs were used to solve Remote Associate Test [81]. However, not much work has been done for creating novel concepts by exploiting graph structures, which is a promising direction for future research.

6.1.5 Graph-based Representation of Document

In the area of text summarization, the units of meaning in a given document are built into a graph (based on the relations extant between them), which is considered as a semantic representation of the document. Whereas it may be excessive to consider these representations as atomic concepts, they are clearly conceptual representations that have high applicability in practical contexts.

To address the problem of identifying salient sentences in biomedical texts, concepts and relations derived from the Unified Medical Language System (UMLS)\(^\text{19}\) have been used to construct a semantic graph that represents a document. The construction process has the following steps: document preprocessing, concept recognition, sentence representation, document representation, and concept clustering [162].

(a) Document preprocessing removes parts of the document that are considered irrelevant for inclusion in the summary, expands acronyms and abbreviations, and splits the content of

\(^{19}\)\url{http://www.nlm.nih.gov/research/umls/}.
the body section into sentences. The preprocessing step can be easily configured to deal with documents of different structures.

(b) **Concept recognition** maps the document text to concepts from the UMLS Metathesaurus\(^\text{20}\) and semantic types from the UMLS Semantic Network\(^\text{21}\). The MetaMap program\(^\text{22}\) is used for this purpose. Concepts from very generic UMLS semantic types are discarded because they have been found to be excessively broad.

(c) **Sentence representation** For each sentence in the document, the UMLS concepts returned by MetaMap are retrieved from the UMLS Metathesaurus along with their complete hierarchy of hypernyms (*is_a* relations). All the hierarchies for each sentence are merged, creating a sentence graph where the edges (temporally unlabelled) represent semantic relations, and only a single vertex is created for each distinct concept in the text. Finally, the two upper levels of this hierarchy are removed, again because they represent very general concepts.

(d) **Document representation** Next, all the sentence graphs are merged into a single document graph. This graph can be extended using more specific relations between nodes to obtain a more complete representation of the document: the *associated with* relation between semantic types from the UMLS Semantic Network or the *related to* relation between concepts from the UMLS Metathesaurus. To expand the document graph, only relations that link leaf vertices are added. Then, each edge of the document graph is assigned a weight between 0 and 1, attaching greater importance to specific concepts than to general ones.

(e) **Concept clustering** groups the UMLS concepts in the document graph using a degree-based clustering algorithm similar to the one proposed in [240]. The aim is to construct sets or clusters of concepts that are closely related in meaning, under the assumption that each cluster represents a different topic in the document and that the most central concepts in the clusters (the centroids) give the necessary and sufficient information related to each topic.

To complete the summarization task two more steps are necessary: sentence-to-cluster assignment and sentence selection. The selection of sentences for the summary is based on the presence in them of the most representative concepts for each topic.

The above method has been applied in biomedical summarization [162, 163], news summarization [161], multidocument summarization of tourism websites [159], retrieval of electronic health records [160] and sentiment analysis [33].


6.1.6 Plan Operator

A different type of concepts (related with the implementation of narrative systems but also with the much broader field of planning as considered in artificial intelligence [120]) are actions as operators that change the world. In the field of planning, such actions are defined as plan operators. Actions in a story are applicable if certain conditions hold in the state of the world before they happen, and after they happen they change the state of the world. This idea has been represented by defining actions with an associated set of preconditions and another of postconditions or effects. Figure 15 shows examples of story actions linked by preconditions.

Specific planners [63, 155] may represent planning operators in different ways. Attempts have been made to standardise AI planning languages, with the Planning Domain Definition Language (PDDL) [133] being a significant reference in this effort. The problem of constructing plan operators for specific applications is an open research question, with ongoing efforts [38] considering constructing them as a composition of general components. Concept creation technologies could be applied in this case with considerable advantage.

![Figure 15: Examples of story actions.](image)

6.1.7 Narratological Category

An expanding area of research has emerged in recent time around computational narratology. This involves computational representation and treatment of the fundamental ingredients of narratives. This effort has been based on existing analyses by literary scholars of the ingredients of narrative, and it has lead to various proposals for their computational representation. Of the many theories of narrative developed in the Humanities, only a few have bridged the gap to become tools in the hands of AI researchers. Of these, Propp’s Morphology of the Folktale [165], is the most extended, having been applied in several AI systems for story generation. The two corner stones of Propp’s analysis of Russian folk tales are a set of roles for characters in the narrative (which he refers to as dramatis personae), and a set of character functions, understood as acts of the character, defined from the point of view of its significance for the course of the action. These have been used in several systems [61, 72, 79, 107, 217, 229], represented in slightly different ways. Of all these, the most explicit conceptual representation of Propp’s set of narratological categories is the description logic formulation developed by Peinado [156]. At a different level of detail, another favourite is the three-act restorative
structure. This model, derived from Joseph Campbell’s analysis of the structure of myths [30], which is a dominant formula for structuring narrative in commercial cinema [228]. Another source that is also being considered in AI is the work of Chatman [34]. This model constitutes a step up from the models of Propp or Campbell in the sense it considers a wider range of media, from literature to film. From the point of view of the AI researcher in search of a model, the greatest advantage of Chatman’s approach is his effort to identify a common core of elementary artefacts involved in several approaches to narrative theory. Chatman studied the distinction between story and discourse, and proposed ways of decomposing each of these domains into elementary units. His idea of structuring story and discourse in terms of nuclei and attached satellites provides a very good way of organising internally the knowledge entities that computational systems rely on for conceptual representation.

A very interesting detail is the fact that there are ongoing research efforts to learn automatically the equivalent of Propp’s morphology from a set of annotated texts [64]. This process involves the automatic creation of the set of character functions as new concepts. This is achieved by Analogical Story Merging (ASM), a novel machine learning technique which provides computational purchase on the problem of identifying a set of plot patterns from a given set of stories. Propp’s manner of abstracting narrative structure from a set of stories is far from being the only possible one. Concept creation technologies should consider possible alternative abstractions which might be automatically generated from a corpus of samples of stories.

6.2 Music

According to Honing [99], music representations are either predominantly technical, that is, are designed in response to a particular technical problem, or else aim to represent conceptual or mental musical structures. The former category emphasises observable and measurable musical attributes, such as the key press of a piano keyboard, or the position of a note head on a musical score. The latter seeks to capture aspects of the listening experience of music, predominantly as part of a computational theory aiming to predict aspects of musical behaviour.

In an early paper discussing the application of computer technology to music research, Babbitt [8] employs the terms acoustic, graphemic, and auditory to distinguish three related domains of musical information. The acoustic domain encompasses the physical manifestations of music, such as the propagation of sound waves, and representations of such properties, such as the analogue representation of music stored on electromagnetic tape, or the stream of bits resulting from analogue-to-digital conversion. The representation of acoustic information most naturally falls within Honing’s category of technical representations.

The graphemic domain pertains to the graphical notation of music, such as conventional musical scores and tablature. Graphical notations are themselves representations of music, serving primarily as musical aide-mémoires and for the communication of musical
ideas. From the computational perspective, there is scope here for both technical and cognitive representational approaches. Where the aim is simply to represent the exact layout of notation symbols on a score, a purely technical representation is adequate. However, if the aim is to also represent associated music-theoretical meaning, or possible performance interpretations, then the representation language must necessarily express, at least in part, the musical knowledge assumed by each notation system. Such information could also be described as declarative knowledge or procedural knowledge. For example, the representation could describe a trill declaratively as an object of ornamentation, or alternatively, as a form of procedural knowledge describing how the trill is created [99].

The auditory domain covers information about music as perceived by the listener, aligning with Honing’s category of conceptual and mental representations. The characterisation of musical information into the domains of the acoustic, graphemic, and auditory is not exhaustive; for example, gestures made by performers would be another potentially relevant domain of information [196]. However, the distinctions are nonetheless important categories of musical information. The phenomenon of music itself cannot be said to exist in any one domain exclusively, but instead can be understood as something that exists between the domains, with each one offering a particular perspective from which to study music [233].

Huron [105] advocated a goal-oriented approach to music representation, in which the question of what aspects of music to represent is largely determined by the intended application. However, one could argue that for an application whose intended purpose is to allow the user to relate information across different domains of musical information, such as between graphemic and acoustic information, then a certain degree of generality is beneficial, if not necessary, to facilitate such mappings. Further problems can also be identified in taking a strictly goal-oriented approach to music representation. Defining a representation for one specific task potentially limits the sharing of data and tools, as inevitably information considered irrelevant for one task will be necessary for another. This is perhaps not a serious issue for individual researchers, but is potentially inhibiting of progress across the research field as a whole. Furthermore, it is impossible to anticipate all possible situations in which users might want to use a representation, and indeed Huron states extensibility as one of the qualities of a good representation. To facilitate user extensibility, it is necessary to employ a suitably general framework within which extensions can be defined within the semantics of the representation itself.

6.2.1 Charm: A General Representation of Music

A simple, yet powerful approach to a general representation of music is proposed by Wiggins et al. [234], Harris et al. [85], and Smaill et al. [201]. The Common Hierarchical Abstract Representation for Music (Charm) aims to support a high degree of generality within the representation itself, which would ultimately require a representation capable of encompassing all possible mental, physical, and cultural manifestations of music and musical behaviour.
Charm is defined initially as a representation of music at the symbolic level, in which identifiable aspects of music are represented by discrete symbols. As such, Charm is appropriate for representing a wide range of graphemic information, but may appear less appropriate for acoustic or other continuously-valued musical information. However, as a general framework for musical representation, developing Charm compliant representation of acoustic and auditory information is perfectly feasible. Symbolic representations are particularly appropriate for the high-level description of a range of perceptual attributes and concepts, such as for representing discrete musical events, groupings of events, and for expressing the formal properties of relationships between such structures.

Charm is based on the computer science concept of abstract data typing. The authors noted that despite the direct incompatibility of many music representation schemes, that a considerable degree of commonality exists at an abstract level. For example, most schemes define some way of representing pitch, whether in terms of MIDI note numbers, scale degree, microtonal divisions of the octave, or frequency. However, at an abstract level, common patterns of operations can be observed, which are irrespective of the underlying implementation. Therefore, the authors proposed an abstract representation, in which musically meaningful operations can be defined in terms of abstract data types. Harris et al. [85] defined basic data types for pitch (and pitch interval), time (and duration), amplitude (and relative amplitude) and timbre. Therefore, the abstract event representation is the Cartesian product:

$$Pitch \times Time \times Duration \times Amplitude \times Timbre$$

In the case of time, the following functions can be defined where the arguments \(\{t, d\}\) denote Time or Duration data types respectively.

$$add_{dd} : Duration \times Duration \rightarrow Duration$$
$$add_{td} : Time \times Duration \rightarrow Time$$
$$sub_{tt} : Time \times Time \rightarrow Duration$$
$$sub_{dd} : Duration \times Duration \rightarrow Duration$$

Typed equivalents of arithmetic relational operators (e.g., \(\leq, \geq, =, \neq\)) are also defined, permitting ordering and equality relations to be determined. With the exception of timbre, the internal structure of each basic data type is the same, allowing comparable functions to be defined modulo renaming [85].

Given a specification of abstract musical data types, a user can implement specific functionality required for their data and application according to the abstract definitions. Implementing specific functionality effectively means supplying a concrete implementation for each operation defined on each abstract type. Or better still, given a suitable high-level language capable of expressing the type information, one could declaratively specify concrete types, with the additional benefit of being able to infer mappings between types.

The abstract data type approach to representing music extends beyond the represen-
tation of surface level events. Charm formally defines the concept of the *constituent*, which allows arbitrary hierarchical structures to be specified [85]. At the abstract level, a constituent is defined as the tuple:

$$\langle \text{Properties}/\text{Definition}, \text{Particles} \rangle$$

*Particles* is a set whose elements, called particles, are either events or other constituents. No constituent can be a particle of itself, defining a structure of constituents as a directed acyclic graph. *Properties/Definition* is the “logical specification of the relationship between the particles of this constituent in terms of the membership of some class” [85]. The distinction between *Properties* and *Definition* is made explicit in a concrete implementation. However, at the abstract level, they both logically describe the structure of the constituent. Properties refer to “propositions which are derivably true of a constituent” [85]; for example, that no particle starts between the beginning and end of any other particle, defined by [85] as a stream:

$$\text{stream} \iff \forall p_1 \in \text{particles}, \exists p_2 \in \text{particles},
\begin{align*}
p_1 &\neq p_2 \land 
GetTime(p_1) &\leq GetTime(p_2) \land 
GetTime(p_2) &< \text{add}(\text{GetTime}(p_1), \text{GetDuration}(p_1))
\end{align*}$$

where *GetTime* and *GetDuration* are selector functions returning the timepoint and duration respectively of a given particle. Definitions are propositions that are true by definition; for example, that a set of particles contains all the events notated in a score of a particular piece of music.

An implementation of a Charm-compliant representation requires some additional properties, both for computational efficiency and user convenience. The following is an example of a simple 'motif' constituent [201].

$$\text{constituent}(c_0, \text{stream}(0, t_1), \text{motif}, [e_1, e_2, e_3, e_4])$$

Every event and constituent defined within the system must be associated with a unique identifier, shown as $c_0$, $e_0$, $e_1$ and so forth in the above example. The constituent is a stream, with a start time and a duration, denoted by the property $\text{stream}(0, t_1)$, which is derivably true from the events it contains. In contrast, the constituent is defined as a motif, and a user is free to provide such definitions for their own purposes.

A wider benefit of adopting an abstract data type approach to music representation is that it provides the basis for developing a common platform for the sharing of data, as well as software tools. This is demonstrated in [201] in which both the implementation language and the concrete representations of the data are shown to be immaterial given that the correct behaviour of the abstract data types is observed. From a formal perspective, many issues of representation discussed in the field can be seen as concerning merely arbitrary matters of encoding or data serialisation. Although encoding schemes may well be designed to meet
particular needs, such as to facilitate efficient human data entry or to be space efficient, the ontological commitments implicit in the encoding can be left unstated, and therefore potentially ambiguous, or even unquestioned, ultimately limiting potential usefulness.

6.2.2 Music Representations in Evolutionary Computation (EC)

Much work on music generation has come from the Evolutionary Computation (EC) community. The music representations used in EC are mostly graphemic [8].

Biles [20] developed one of the first EC approaches in the domain of music, GenJam, which produces Jazz improvisations through a Genetic Algorithms (GA) using a “two-level, position-based, binary representation scheme”. In other words, there are two populations, one of phrases and the other of measures. An individual of the phrases population represents a sequence of MIDI events. An individual of the measures population represents a sequence of phrase individuals through their indexes.

Horner and Ayers [102] applied EC techniques to the harmonisation problem. Their method begins by enumerating all possible correct individual chords and then uses a GA to evolve sequences of these chords. The fixed length strings are used for representation purposes. Wiggins et al. [232] used a GA to evolve harmonisations of chorale melodies and instrumental solos. The representation used to evolve harmonisations consists of four fixed length integer strings encoding the notes of the melody (degree and duration). These correspond to the soprano, alto, tenor and bass parts. The soprano part is an input of the user and is not subjected to evolution. Solos are represented by a string of (degree, duration) pairs.

Ralley [168] used GAs to evolve simple melodies. The representation uses two integer strings, a header and a list of melodic events. The header defines the starting note and its index within a scale. The list of melodic events defines the notes which are represented as offsets from the starting note.

The generation of rhythmic patterns has been explored by Horowitz [104] and Pazos et al. [152], using as representation fixed length integer or binary strings. For the same purpose Kaliakatsos-Papakostas et al. [111] evolved a matrix representation, with each row corresponding to the activity of a drum element, and each column representing a certain subdivision of a music measure. The number of drum elements determines the number of rows, while the number of measure sub-divisions determines the number of columns in the rhythm matrix. Cells of the matrix store integers, which encode the intensity of the element at that moment.

6.3 Image

A straightforward representation of an image is a matrix of pixels. Each pixel needs a few bytes to contain color values, e.g., three bytes for RGB images.
den Heijer and Eiben [52] used Genetic Algorithms (GAs) to evolve Scalable Vector Graphics (SVGs), manipulating directly SVG files through a set of specifically designed mutation and recombination operators. In a more recent approach den Heijer [51] manipulate directly BMP, GIF, PNG and JPG files to produce glitch art effects (Fig. 16).

![Glitch art images](image)

Figure 16: Glitch art images [51]. Reproduced by permission.

We consider *parametric representations* as a particular type of descriptive representation where the representation encodes a set of parameters that determine the behaviour of a generative system. The work of Draves [54] is a prototypical example of such approach, where the genotype encodes the parameter set of fractal flames, i.e., a set of up to several hundred floating-point numbers. In the words of Draves:

*The language is intended to be abstract, expressive, and robust. Abstract means that the codes are small relative to the images. Expressive means that a variety of images can be drawn. And robust means that useful codes are easy to find.*

Other recent examples include the work of Reed [173] who evolved a bezier curve that is then rotated around an axis to create a vase. The representation consists of five coordinates and three integers, which determine the angles within the curve. Machado and Amaro [126] used a string of floating-point numbers to encode and evolve a set of parameters that specify the sensory organs and behavior of artificial ants, who are used to create non-photorealistic renderings of input images. Fornari [66] used an integer string to evolve the behavior of birdsong generation engine.
6.4 Emotion

Emotions have become in recent times a very interesting focus of interest in computational applications. There are different representations of emotions [43] but two of them are the most often used: emotional categories and emotional dimensions.

**Emotional categories** Natural languages provide assorted words with varying degrees of expressiveness for describing emotional states. Several approaches have been proposed to reduce the number of words used to identify emotions, such as basic emotions, super-ordinate emotional categories and essential everyday emotion terms. Basic emotions refer to those that are more well-known and understandable to everybody than others [43]. In the super-ordinate emotional categories, some emotional categories are proposed as more fundamental, with the argument that they subsume the others [193]. Finally, the essential everyday emotion terms focus on emotional words that play an important role in everyday life [44].

**Emotional dimensions** are spatial representations, which model the essential aspects of emotions numerically. They deal with artificially imposed scales of identified characteristics of emotions. Although there are different dimensional models with different dimensions and numerical scales [65], most of them agree on three basic dimensions called evaluation, activation and power [149]. Evaluation represents how positive or negative an emotion is. At one extreme we have emotions such as happiness, satisfaction and hope while at the other we find emotions such as unhappiness, dissatisfaction and despair. Activation represents an activity versus passivity scale of emotions, with emotions such as excitation at one extreme, and emotions such as calmness and relaxation at the other end. Power represents the sense of control which the emotion exerts on the subject. At one end of the scale we have emotions characterized as completely controlled, such as fear and submission and at the other end we find emotions such as dominance and contempt. Emotional dimensions describe a continuous space as opposed to the discrete space of emotional categories. To assess the three dimensions, there is an affective rating system originally devised by Lang [119] called SAM. The graphic SAM figures comprise bipolar scales that depict different values along each emotional dimension. For the evaluation dimension SAM ranges from a smiling, happy figure to a frowning, unhappy figure; for the activation dimension SAM ranges from an excited, wide-eyed figure to a relaxed, sleepy figure; for the power dimension, SAM ranges from a small figure (dominated) to a large figure (in control). An experiment subject can select any of the five figures on each scale or the space between two figures, which results in a 9-point rating scale for each dimension. Besides, there are collections of evaluation ratings estimated by computational methods, called affect lexicons, such as General Inquirer, WordNet-AFFECT [207], SentiWordNet [9, 58], Macquarie Semantic Orientation Lexicon (MSOL) [142] and SenticNet.

---

23 http://www.wjh.harvard.edu/~inquirer/.
24 http://sentic.net.
7 Conclusions and Future Work

As this review shows, there are numerous ways of representing information for accomplish-
ing various tasks. We organized the conceptual representations reviewed in accordance with
two important perspectives on the distinctions between them. One distinction is between
symbolic, spatial, and connectionist representations. The other is between descriptive and
procedural representations. These two distinctions are mutually orthogonal. We hope that
this organization would act as a map, helping researchers navigate in the forest of conceptual
representations. In addition, we recognized the different levels of generality of conceptual
representations regarding domains. Some representations are applicable to many domains,
whereas some are tailored for the information of specific domains. Domain-generic represen-
tations are especially useful in applications that process information from multiple domains.

This review demonstrates that conceptual representations are abundant at each of the
symbolic, spatial, and connectionist levels. Still, promising new representations, at every
level, are emerging, such as bisociation (Section 2.5), heterogeneous information network
(Section 2.6), conceptual spaces (Section 3.1), neural assembly (Section 4.2), and deep rep-
resentation (Section 4.3). Moreover, in the literature, descriptive representations dominate,
over procedural representations. Would cautiously seeking procedural representations provide
opportunities of concept creation?

ConCreTe will use various conceptual representations in concept creation tasks. Broadly
speaking, concept formation can be roughly divided into four subcategories, concept extrac-
tion, concept learning, concept discovery, and concept creation. We also point out some
specific topics of future research, as identified during this review, when describing each of the
subcategories.

Concept extraction is the task of extracting or transforming a conceptual representation
from an existing representation. Concept extraction is frequently applied to textual corpora,
such as in extraction of semantic relations. Interesting future work of this direction are
constructing bisociative triplets and automatically building plan operators.

Concept learning is a supervised activity, where a description of an existing concept is
formed inductively from (positive and negative) examples of the concept. In logical terms,
the extension of the concept is given and machine learning is then used to construct the
intension. The representation can be essentially any of the ones reviewed above, and in the
review we included indications of how concept descriptions can be learned.

Concept discovery is an unsupervised inductive activity where the concepts are not known
in advance, but methods such as clustering are used to identify natural concepts from given
examples. Again, many of the representations reviewed above are applicable. Two par-
ticularly interesting future work are discovering new syntactic categories and learning new
narratological categories.
**Concept creation** is the task we are mostly interested in in the ConCreTe project: how to create novel, meaningful concepts. Extending the typology of Boden [23], at least the following approaches can be used to create new concepts.

(a) *Mutational creativity*: modify some given concept by adding, removing or changing something in it. Two specific operations are *generalization* and *specialization* of existing concepts.

(b) *Combinational creativity*: combine two or more existing concepts to form a new one, such as in *conceptual blending*. In the review above, we also mentioned that ANNs (Artificial Neural Networks) can be used to combine or create random variations of learned patterns.

(c) *Exploratory creativity*: assuming a space of potential creative artifacts, defined either declaratively or procedurally, explore the space for novel and interesting concepts. An opportunity for exploratory creativity is discovering novel concepts in word association graphs by exploiting graph structure measures.

(d) *Transformational or meta-creativity*: any of the above where additionally the rules, assumptions or goals are also modified. For example, finding new interpretations of familiar concepts in a conceptual space model.

In summary, the conceptual representations reviewed are the basis of our ongoing work on concept creation, which is the main focus of ConCreTe. Inevitably, we will also contribute to other three tightly connected tasks, concept extraction, concept learning, and concept discovery.
References


45


[97] J. Hoffart, F. M. Suchanek, K. Berberich, E. Lewis-Kelham, G. de Melo, and G. Weikum. YAGO2: exploring and querying world knowledge in time, space, con-


[123] D. Lin and P. Pantel. DIRT @SBT@Discovery of Inference Rules from Text. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’01, pages 323–328, New York, NY, USA, 2001. ACM.


[163] L. Plaza, M. Stevenson, and A. Díaz. Resolving ambiguity in biomedical text to improve


