

# Automatic Creativity Measurement in Scratch Programs Across Modalities

Anastasia Kovalkov<sup>1</sup>, Benjamin Paaßen<sup>2,3</sup>, Avi Segal<sup>1</sup>, Niels Pinkwart<sup>2,3</sup>, and  
Kobi Gal<sup>1,4</sup>

<sup>1</sup>Ben-Gurion University of the Negev

<sup>2</sup>Humboldt University of Berlin

<sup>3</sup>German Research Center for Artificial Intelligence

<sup>4</sup>University of Edinburgh

©2021 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

## Abstract

Promoting creativity is considered an important goal of education, but creativity is notoriously hard to measure. In this paper, we make the journey from defining a formal measure of creativity that is efficiently computable to applying the measure in a practical domain. The measure is general and relies on core theoretical concepts in creativity theory, namely fluency, flexibility, and originality, integrating with prior cognitive science literature. We adapted the general measure for projects in the popular visual programming language Scratch. We designed a machine learning model for predicting the creativity of Scratch projects, trained and evaluated on human expert creativity assessments in an extensive user study. Our results show that opinions about creativity in Scratch varied widely across experts. The automatic creativity assessment aligned with the assessment of the human experts more than the experts agreed with each other. This is a first step in providing computational models for measuring creativity that can be applied to educational technologies, and to scale up the benefit of creativity education in schools.

**Keywords:** Creativity, distances, Scratch, computer science education, automatic assessment tools

## 1 Introduction

Creativity has been shown to promote students' critical thinking, self-motivation, and mastery of skills and concepts Henriksen et al. [2016], Knobelsdorf and Romeike [2008], Resnick et al. [2017]. To track student's creative achievement, we require instruments to quantify creativity. However, creativity is notoriously hard to quantify because it is highly context-dependent Amabile [2018], Henriksen et al. [2016]. Amabile [2018] addressed this challenge by relying on a panel of domain experts who assess each creative product. Unfortunately, it is hardly feasible to ask an expert panel to rate the creative products of millions of students. This begs

the question whether it is possible to construct an automated creativity assessment technique that can serve as a surrogate for expert assessment.

Prior attempts at automating creativity assessment include Huang et al. [2010], Kovalkov et al. [2020], Yeh and Lin [2015]. All these approaches build upon creativity tests from the psychometric literature Torrance [1972], Williams [1979], Runco et al. [2016], in particular the fluency, flexibility, and originality scales from Torrance’s test of creative thinking Torrance [1972]. Fluency refers to the number of generated ideas, flexibility to the number of distinct classes of generated ideas, and originality to the infrequency of ideas compared to a typical population. Prior approaches have implemented automatic versions of these scales for specific domains Huang et al. [2010], Kovalkov et al. [2020], Yeh and Lin [2015].

In this paper, we propose a novel formalization of fluency, flexibility, and originality that is flexible enough to be adapted across modalities while remaining efficient to compute. Our formalization requires two ingredients: A set of concepts for each modality, and a distance between these concepts. Then, we can treat any creative product as a structured combination of concepts from each modality. We measure fluency as the distance to an empty product, flexibility as the distance between concepts in the product, and originality as the average distance to typical products. We show that our measure is a proper generalization of Torrance’s scales Torrance [1972], that is, we obtain Torrance’s notions of fluency, flexibility, and originality as a special case for a certain distance between concepts.

Note that our formalization is designed to be general. To demonstrate its practical feasibility, we implement it for a specific multimodal domain, namely Scratch, a visual programming environment designed for open-ended, creative learning Maloney et al. [2010]. In doing so, we expand our prior work which considered only code and images Kovalkov et al. [2021]. Scratch is particularly interesting for automated assessment because many Scratch students program on their own without access to teacher feedback, such that an automatic feedback system would benefit them in particular.

We conducted a user study to collect expert assessments of creativity of Scratch projects. The experts were Scratch instructors without prior knowledge in creativity theory. Each expert was assigned a set of preselected Scratch projects and asked to separately assess the creativity of projects according to four different aspects: code, visuals, audio, and idea behind the project.

We designed an online application to facilitate the assessment process, which allowed the experts to interact with each project as needed. We compared the resulting expert assessments to our automatic assessments. We find that the automatic assessments agree with expert assessments at least as much as experts agree with each other.

The contribution of this work is twofold. First, we provide a novel and general computational framework to measure creativity, which extends classic notions of creativity in the literature. Second, we implement this framework for Scratch and thus provide an automatic measure to scale up creativity assessment.

## 2 Related Work

The largest body of prior work on creativity stems from psychology, where creativity is considered a crucial aspect of the human intellect Runco and Jaeger [2012]. Guilford’s model on the structure of intellect includes creativity as *divergent production*, meaning the ability to generate a wide variety of ideas on the same topic Guilford [1956]. Based on this definition of creativity, creativity tests have been developed, which present participants with a prompt and ask for as many ideas as possible in reaction to that prompt Torrance [1972], Williams [1979], Runco et al. [2016], Kim [2006].

Consider the example in Fig. 1. Here, the prompt consists of three geometric shapes and the task is to generate as many objects from these shapes as possible. In Torrance’s test of creative thinking Torrance [1972], we then count the number of generated objects and call this number *fluency*; we call the number of distinct classes of objects *flexibility*, and we call the infrequency of objects compared to a typical population of participants *originality* Torrance [1972]. In this work, we focus particularly on these three scales from Torrance’s test as prior research has already established connections between Torrance’s scales and success in beginner’s programming Hershkovitz et al. [2019], Israel-Fishelson et al. [2021].

A challenge of classic creativity tests is that we need human experts to count the number of generated ideas and decide which ones belong to the same class. This reliance on human labor becomes infeasible in scenarios where a large number of people engages in creative activity, such as large-scale open learning environments like Scratch Maloney et al. [2010]. The problem is compounded by the fact that a single creativity measurement is likely not enough, given that creativity is context-dependent and, thus, changes over time Amabile [2018]. Accordingly, our mission in this paper is to adapt the definition of fluency, flexibility, and originality in two respects: First, by making it applicable to student submissions in a multimodal learning environment, such as Scratch. Second, by making it efficiently computable, without the need for human intervention.

There exists some prior work on automating Torrance’s test for online learning. In particular, Huang et al. [2010] compute fluency, flexibility, and originality in a collaborative brainstorming task; Yeh and Lin [2015] automatically count the number of unique ideas in reaction to an inkblot-like picture; and Kovalkov et al. [2020, 2021] automatically grade the creativity of Scratch projects with a manually defined measure. Our present work is a generalization of these prior approaches by formalizing fluency, flexibility, and originality abstractly, based on concepts and distances between them. This abstraction has the advantage that we can transfer conceptual and computational approaches between modalities or even domains with little need for adaptation.

The reason we use distances is twofold. First, distances are a common representation in data mining and there is a rich toolbox of distance measures which we can apply for our purposes Pękalska and Duin [2005]. More importantly, distances are helpful to model the organization of knowledge in the mind. In particular, we can say that two concepts have a low distance to each other if humans tend to associate them more easily Hodgetts et al. [2009], Kenett [2019]. Two popular frameworks in this regard are *semantic embeddings* and *semantic networks*. Semantic embeddings assume that concepts are implicitly represented in a vector space and that their distance in this space corresponds to their semantic relatedness Kenett [2019], Landauer et al. [1998]. By contrast, semantic networks assume that concepts are organized as a graph and that the shortest path distance in the graph represents semantic relatedness Kenett [2019], Boden [2004], Georgiev and Georgiev [2018], Sowa [2006]. Kenett [2019] suggests that both frameworks play a role in quantifying creativity by providing complementary notions of distance between concepts. We show later that our formalization is general enough to accommodate both frameworks.

We test our formalization in the example domain of Scratch. Scratch is an interactive, block-based, and graphics-focused programming environment used for introductory programming Maloney et al. [2010]. Scratch is particularly interesting for us because Scratch projects are intended for creative expression Bustillo and Garaizar [2016], Giannakos et al. [2013] and involve components from three different modalities, namely code, visuals, and audio. Our ambition is to develop an automatic measure of creativity that can be applied to all three modalities, whereas prior work has either disregarded image and sound entirely Hershkovitz et al. [2019] or used only an ad-hoc definition without theoretical justification Kovalkov et al.

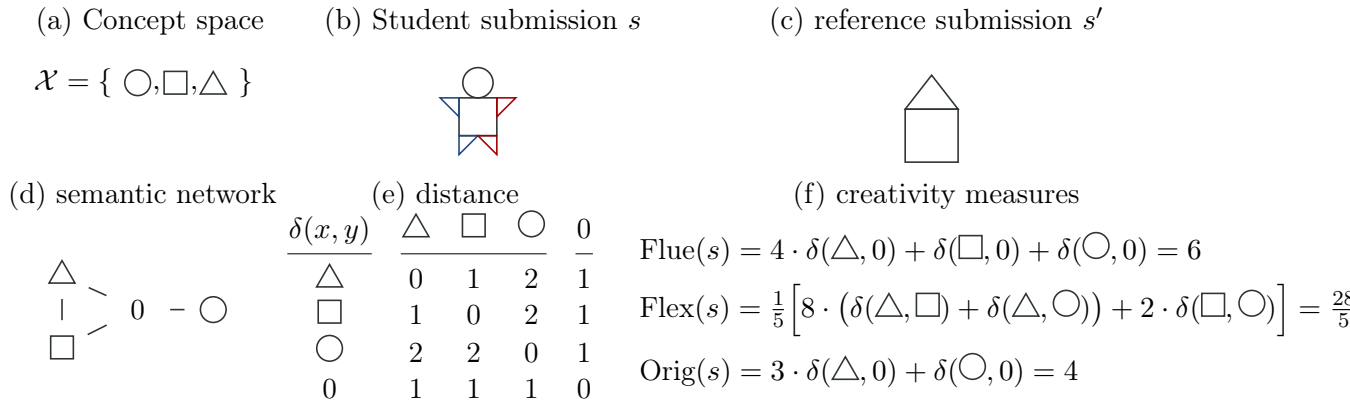


Figure 1: Example creativity task illustrating our proposed measure of creativity. (a) Participants receive a set of three geometric shapes (triangle, rectangle, and circle) as stimulus and have the task to form objects from these shapes. (b) Example answer (a figure shaped out of a rectangle, four triangles, and a circle; color is used to disambiguate triangles). (c) Less creative answer (a house out of a rectangle and a triangle) as reference for originality. (d) Possible semantic network including the triangle, square, and circle shape. (e) Possible definition of  $\delta$  for this task based on the path distance in a semantic network. (f) Fluency, flexibility, and originality of the figure (b) according to  $\delta$ .

[2020].

### 3 Formal Creativity Measure

In this section, we introduce our proposed measure of creativity. Our main inspiration is Torrance’s test of creative thinking, which quantifies creativity by counting the number of ideas in a creative product (fluency), the number of unique classes of ideas (flexibility), and the infrequency of ideas compared to a reference population (originality) Torrance [1972], Kim [2006]. We use these three scales since prior work has already shown that they lend themselves for automation in learning environments Huang et al. [2010], Kovalkov et al. [2020], Yeh and Lin [2015] and are connected to success in beginner’s programming Hershkovitz et al. [2019], Israel-Fishelson et al. [2021]. However, there are domains where a simple counting scheme may be insufficient. Consider the example of Scratch projects (Fig. 2). A Scratch project consists of code, images, and sounds. If we merely count the number of code blocks, images, and sounds, we would lose a lot of information that could give us additional insight into the creativity of the project, for example, whether the code used advanced programming concepts or repeated many simple operations. Instead, we propose to measure the difference between programming concepts (and concepts more generally) by a distance metric. This distance, then, is the basis for our formalizations of fluency, flexibility, and originality.

In general, our measure of creativity (Section 3.3) relies on two ingredients, which we describe next: The concept space (Section 3.1) in which creative products live, and a distance between concepts (Section 3.2).

### 3.1 Concept Spaces and Creative Products

We define a creative task as providing a student with a set of building blocks, which the student can combine to a creative product. We call these building blocks *concepts* and the set of all available concepts the *concept space*  $\mathcal{X}$ . Consider the example in Fig. 1. Here, the creative task is to combine the three shapes circle, square, and triangle in a meaningful way. Accordingly, the concept space is  $\mathcal{X} = \{\bigcirc, \square, \triangle\}$ .

Next, we define a *creative product*  $s$  as a combination of concepts. More precisely, we say that  $s$  is a graph  $s = (V_s, E_s)$ , where  $V_s \subseteq \mathcal{X}$  are the concepts in  $s$  and where  $E_s \subseteq V_s \times V_s$  are the (syntactic) connections between concepts in  $s$ . For example, the house in Fig. 1(c) would be represented as the graph  $s' = (\{\square, \triangle\}, \{(\square, \triangle)\})$ , because it involves a square and a triangle, where the square is connected to the triangle. Similarly, the figure in Fig. 1(b) would be represented as the graph  $s = (\{\bigcirc, \square, \color{blue}{\triangleleft}, \color{blue}{\triangleright}, \color{red}{\triangleleft}, \color{red}{\triangleright}\}, \{(\bigcirc, \square), (\color{blue}{\triangleleft}, \square), (\color{blue}{\triangleright}, \square), (\color{red}{\triangleleft}, \square), (\color{red}{\triangleright}, \square)\})$ .

Prior literature has shown that a graph representation is flexible enough to represent student work in a wide variety of domains and modalities Mokbel et al. [2013]. For example, we can represent computer programs by syntax trees, where the structure of the code is represented by the connections within the trees Paaßen et al. [2018], Price et al. [2017], Rivers and Koedinger [2017]; we can represent images as a collection of depicted objects and their spatial relation via connections Johnson et al. [2015]; and we can represent audio data as a sequence of sounds, where connections represent the temporal ordering.

Importantly, to automate creativity computation, we require an automatic way to convert raw student output into a graph of concepts. Fortunately, such a conversion is natural for many domains Mokbel et al. [2013]. For example, we describe the conversion for Scratch projects in Section 4.

### 3.2 Distances

As a next step, we formalize the semantic relatedness between concepts in a domain by means of a distance function  $\delta$ . More precisely, we define a (semantic) distance  $\delta$  as a function that maps two concepts  $x \in \mathcal{X}$  and  $y \in \mathcal{X}$  to a real number, such that  $\delta(x, y) \geq 0$ ,  $\delta(x, y) = \delta(y, x)$ , and  $\delta(x, y) = 0$  if and only if  $x = y$ . These requirements stem from the mathematical definition of a distance metric Pełkalska and Duin [2005]. We use distances as an interface because they are sufficient to define creativity but abstract enough to be easily adaptable across domains and modalities Pełkalska and Duin [2005]. Additionally, distances connect nicely to prior research in cognitive science. To illustrate this connection, we provide two examples of distances.

First, we consider semantic networks Boden [2004], Sowa [2006]. The theory of semantic networks suggests that human cognition arranges concepts in a graph, where connections express semantic relatedness. We can obtain a distance  $\delta$  from a semantic network by setting the concept space  $\mathcal{X}$  to the nodes of the network and  $\delta$  to the minimum number of edges we need to traverse to get from one concept to another Georgiev and Georgiev [2018]. Fig. 1(d) shows a very simple example of such a semantic network. In this network, we connect triangle and square because they are related—for example, we can construct a square out of two triangles and the square is the next regular polygon by number of vertices. By contrast, the circle is not directly related to either shape, which is why we do not connect it directly but only via a 'neutral shape' 0. Accordingly, the distance between triangle and square is  $\delta(\triangle, \square) = 1$ , because these shapes are directly connected in the network, and  $\delta(\triangle, \bigcirc) = 2$  because we need to make two hops to get from the triangle to the circle. The matrix of all pairwise distances is shown in Fig. 1(e). In Fig. 3, we use a semantic network to represent

the relatedness between code blocks in Scratch.

Second, we consider semantic embeddings Kenett [2019], Landauer et al. [1998]. Semantic embeddings assume that concepts have an implicit representation in a high-dimensional vector space and that their semantic relatedness corresponds to the Euclidean distance or the cosine similarity between the vectors. Accordingly, we can set  $\delta$  to the Euclidean distance  $\delta(\vec{x}, \vec{y}) = \|\vec{x} - \vec{y}\|_2$  or the cosine distance  $\delta(\vec{x}, \vec{y}) = 1 - \frac{\vec{x}^T \cdot \vec{y}}{\|\vec{x}\| \cdot \|\vec{y}\|}$ . In Section 4, we use the Euclidean distance for sounds and the cosine distance for images in Scratch.

A final feature of our proposed distance-based framework is that we can express the amount of domain knowledge expressed by a concept via its distance  $\delta(x, 0)$  to a neutral nullconcept 0. For example, in Scratch code we use the distance  $\delta(x, 0)$  to express how advanced a programming concept is (Section 4).

### 3.3 Computing Creativity

In this section, we adapt Torrance’s scales of fluency, flexibility, and originality Torrance [1972] to our formalization. First, let us recall how Torrance’s test works: We present participants with a prompt in form of a concept space  $\mathcal{X}$  and then ask them to generate as many creative products over this concept space as possible. The number of generated creative products is called fluency, the number of distinct classes of generated products is called flexibility, and the infrequency of products in comparison to a general population is called originality.

Our aim here is slightly different: We consider a single creative product and wish to quantify the amount of creativity expressed by this product in terms of fluency, flexibility, and originality. Fortunately, the three scales still apply if we count concepts instead of products. In more detail, we obtain the following three scales.

#### 3.3.1 Fluency

Torrance’s test defines fluency as the number of generated ideas Kim [2006], Torrance [1972]. Applied to a single creative product, this would mean that a product expresses more fluency if it contains more concepts. Thanks to our distance-based framework, we can be more nuanced. We formalize the *amount* of fluency expressed by a concept  $x$  by its distance  $\delta(x, 0)$  to the nullconcept. The fluency of a product is the sum over all these distances within the product:

$$\text{Flue}(s) = \sum_{x \in V_s} \delta(x, 0) \quad (1)$$

Note that this is a proper generalization over counting the number of concepts because we can set  $\delta(x, 0) = 1$  for all concepts  $x$ , yielding  $\text{Flue}(s) = |V_s|$ . Computationally, fluency is efficient because it only requires  $|V_s|$  distance computations.

For example, the fluency of the figure in Fig. 1(b) is 6 because it consists of six shapes, each of which has distance 1 to the nullconcept (Fig. 1(f)).

#### 3.3.2 Flexibility

Torrance’s test defines flexibility as the number of distinct classes of ideas that are generated Kim [2006], Torrance [1972]. In other words, if a participant only generates ideas that belong to the same class—for example, different houses for the drawing task in Fig. 1—then this would be counted as a flexibility of one. Each additional distinct class of ideas increases flexibility by one. According to Kim [2006], this definition is inspired by Guilford’s structure of intellect model Guilford [1956] and is meant to capture the diversity of generated ideas. Fortunately, our framework enables us to capture the diversity of concepts in more nuance

than a binary choice whether it's distinct or not. In particular, we can set the distance  $\delta(x, y)$  such that it quantifies *how* distinct  $x$  is from  $y$ . Once we have set up  $\delta$  in this way, we formalize flexibility as the sum of all pairwise distances between concepts, normalized by a factor of  $|V_s| - 1$  to maintain the same scale as for fluency.

$$\text{Flex}(s) = \frac{1}{|V_s| - 1} \sum_{x \in V_s} \sum_{y \in V_s} \delta(x, y) \quad (2)$$

For flexibility, it makes sense to preprocess  $V_s$  and remove duplicates. This conforms to Torrance's notion that flexibility should represent distinct classes of ideas. With this preprocessing step, flexibility is a proper generalization over counting the number of distinct concepts because we can set  $\delta(x, y) = 1$  if  $x \neq y$  and  $\delta(x, x) = 0$ , yielding a flexibility of  $\text{Flex}(s) = |\tilde{V}_s|$ , where  $\tilde{V}_s$  refers to the duplicate-free version of  $V_s$ . Regarding computational complexity, flexibility requires a sum of pairwise distances, which is in  $\mathcal{O}(|V_s|^2)$ .

For example, the figure in Fig. 1(b) yields a flexibility of  $\frac{28}{5}$ , because it consists of four triangles, one square, and one circle, and we hence add four times the distance  $\delta(\triangle, \square) = 1$ , four times  $\delta(\square, \triangle) = 1$ , four times  $\delta(\triangle, \circ) = 2$ , four times  $\delta(\circ, \triangle) = 2$ , one time  $\delta(\square, \circ) = 2$ , and one time  $\delta(\circ, \square) = 2$ , yielding  $8 \cdot 1 + 10 \cdot 2 = 28$ . Then, we normalize by the number of shapes minus one, yielding  $\frac{28}{5}$  (Fig. 1(f)).

### 3.3.3 Originality

Torrance's test defines originality as the statistical infrequency of ideas in comparison to a general population Kim [2006], Torrance [1972]. In other words, one first needs to calibrate the test using a sample of participants from the general population, yielding a frequency  $f(x)$  for each idea  $x$ . Then, for each new participant, we can determine the *infrequency* of each idea as  $1 - f(x)$  and add up these infrequencies to obtain originality.

A shortcoming of this definition is that it can not distinguish the originality of two ideas which never occurred before. Consider our example in Fig. 1 and let's compare the figure in Fig. 1(b) with a variation of the house in Fig. 1(b), where we just add another square to represent the door of the house. Both shapes, the figure and the altered house, are distinct from the house  $s'$ —but the figure is arguably *more* original because it is less similar to the house. This is in line with Runco and Jaeger [2012] who notes that distance to previously existing ideas can be seen as particular evidence of genius in creative achievement.

Overall, we define the originality of a creative product  $s$  as the average distance  $d_\delta(s, s')$  to typical products  $s'$  in a sample  $S$ .

$$\text{Orig}(s) = \frac{1}{|S|} \sum_{s' \in S} d_\delta(s, s'). \quad (3)$$

Note that, as in Torrance's definition, we require a sample of typical products  $S$  to define originality. Indeed, this appears to be a fundamental property of originality: We always need a reference point with respect to which we define something as original Runco and Jaeger [2012]. For example, Boden [2004] distinguishes between creativity with respect to my own prior ideas and creativity with respect to the entire prior human history. In education, Spendlove [2008] suggests to focus on "ordinary" creativity which can regularly be observed, rather than exceptional genius. Accordingly, the sample  $S$  should be chosen to represent products one would expect of a typical student in the same context. For example, it could be a sample of products from students of a similar age, school system, and socio-economic background. Importantly, we may need different samples to account for different contexts appropriately Amabile [2018].

Another important point is that originality requires a distance  $d_\delta$  between products, which we define next.

### 3.3.4 Distances between products

Given a distance  $\delta$  between concepts, how can we compute a distance  $d_\delta$  between two creative products  $s$  and  $s'$ ? Here, we take inspiration from cognitive science. Hodgetts et al. [2009] reviewed how humans perceive similarity between objects that consist of parts and found two major notions in the literature: Structural alignment and representational distortion.

Structural alignment Markman and Gentner [1993] measures distance by the difficulty of aligning the parts of both objects. For example, consider the figure  $s$  in Fig. 1(b) and the house  $s'$  in Fig. 1(c). We can align the square and one triangle of both objects, but the remaining three triangles and the circle of the figure can not be aligned, meaning that structural alignment would tell us that  $s$  and  $s'$  are fairly dissimilar. However, when we compare the figure  $s$  to itself, we can align every shape, meaning that the distance is zero.

Representational distortion Chater and Hahn [1997] measures distance by the effort needed to transform one object into another by a sequence of predefined transformations. For example, we can transform the figure  $s$  into the house  $s'$  by removing the circle and three triangles and putting the remaining triangle on top of the square.

In general, both structural alignment and representational distortion are hard to compute. Structural alignment requires a search over the combinatorially large space of possible alignments between two objects to find the best one, and representational distortion requires a search over the infinitely large space of possible transformation sequences that turn one object into another. Fortunately, algorithmic research in the past decades has found special cases where these searches become efficient, namely *edit distances* Bille [2005], Bougleux et al. [2017], Levenshtein [1966], Paaßen et al. [2018], Zhang and Shasha [1989].

To define an edit distance, we start from structural alignment. In particular, we define an *alignment* between two products  $s$  and  $s'$  as a set of tuples  $M \subseteq (V_s \cup \{0\}) \times (V_{s'} \cup \{0\})$  such that any concept in  $V_s$  occurs exactly once on the left-hand-side and any concept in  $V_{s'}$  occurs exactly once on the right-hand-side. Consider again the example of the figure  $s$  in Fig. 1(b) and the house  $s'$  in Fig. 1(c). One possible alignment between  $s$  and  $s'$  would be  $M = \{(\blacktriangleright, \triangle), (\blacktriangledown, 0), (\blacktriangleright, 0), (\blacktriangledown, 0), (\square, \square), (\bigcirc, 0)\}$ , that is, we align one of the four limbs of the figure  $\blacktriangleright$  to the roof of the house  $\triangle$  and the torso of the figure  $\square$  to the house's wall  $\square$ . All remaining shapes in  $s$ , namely  $\blacktriangledown$ ,  $\blacktriangleright$ ,  $\blacktriangledown$ , and  $\bigcirc$ , can not be aligned to anything in  $s'$  anymore, such that we must align them to the nullconcept instead. Note that other alignments are possible as well, for example  $M' = \{(\blacktriangleright, \square), (\blacktriangledown, 0), (\blacktriangleright, 0), (\blacktriangledown, 0), (\square, 0), (\bigcirc, \triangle)\}$ , where one of the limbs of the figure is aligned with the wall of the house and the head of the figure is aligned with the roof of the house. Intuitively, the latter alignment is worse because we do not align similar concepts of both products, even though we could. Edit distances follow the same intuition: We define the cost  $c_\delta(M)$  of an alignment  $M$  as the sum over the distances between aligned parts and we define the edit distance  $d_\delta(s, s')$  as the cost of the cheapest alignment between  $s$  and  $s'$ . More formally, we obtain:

$$d_\delta(s, s') = \min_{M \in \mathcal{M}(s, s')} c_\delta(M), \text{ where } c_\delta(M) = \sum_{(x, y) \in M} \delta(x, y), \quad (4)$$

and where  $\mathcal{M}(s, s')$  denotes the set of all alignments between  $s$  and  $s'$ .

Returning to our example above, we see that  $M$  has the cost  $c_\delta(M) = 3 \cdot \delta(\triangle, 0) + \delta(\bigcirc, 0) = 4$  and  $M'$  has the cost  $c_\delta(M') = \delta(\triangle, \square) + 3 \cdot \delta(\triangle, 0) + \delta(\square, 0) + \delta(\bigcirc, \triangle) = 7$ , that is,  $M$  is cheaper than  $M'$ . Indeed,  $M$  is the cheapest possible alignment between  $s$  and  $s'$ , meaning



that  $c_\delta(M) = d_\delta(s, s') = 3$ . Provided that  $s'$  is the only product in our sample  $S$ , this value also expresses the originality of  $s$  (Fig. 1(f)).

It can be shown that the cheapest alignment  $M$  can be found automatically in  $\mathcal{O}((|V_s| + |V_{s'}|)^3)$  using the Hungarian algorithm Munkres [1957]. Even better, it is possible to restrict the set of alignments  $\mathcal{M}$  to take specifics of our data into account. For example, for sound we want to make sure that the alignment preserves the temporal order, yielding the sequence edit distance of Levenshtein [1966] with the complexity  $\mathcal{O}(|V_s| \cdot |V_{s'}|)$ . For code, we want that the tree structure of the program syntax is maintained, yielding the tree edit distance of Zhang and Shasha [1989] with complexity  $\mathcal{O}(|V_s|^2 \cdot |V_{s'}|^2)$ .

Beyond their computational appeal, edit distances have another advantage: they unify structural alignment and representational distortion. This is because we can convert any alignment between two products  $s$  and  $s'$  to a sequence of transformations from  $s$  to  $s'$  and vice versa. In particular, any tuple  $(x, 0) \in M$  corresponds to a deletion of concept  $x$  from  $s$ , any tuple  $(0, y) \in M$  corresponds to an insertion of concept  $y$  into  $s$ , and any tuple  $(x, y) \in M$  corresponds to a replacement of concept  $x$  with concept  $y$ . In this translation, costs are preserved, such that the cheapest alignment also corresponds to the cheapest sequence of deletions, insertions, and replacements between  $s$  and  $s'$  Bougleux et al. [2017], Zhang and Shasha [1989]. Hence, structural alignment and representational distortion become equivalent.

We note that we are not the first to apply edit distances on educational data. Edit distances are a typical tool to compare computer programs Paaßen et al. [2018], Price et al. [2017], Rivers and Koedinger [2017] and have previously been suggested as a domain-independent measure of proximity for educational data Mokbel et al. [2013]. To us, this is additional encouragement that edit distances are a natural way to compute originality in educational settings.

Finally, we note that 3 is a proper generalization of Torrance’s definition in terms of infrequency. In particular, we recover infrequency as a special case by setting  $d_\delta$  to the alignment distance in 4, and by setting  $\delta(x, x) = \delta(0, x) = 0$  with  $\delta(x, y) = 1$ , otherwise. For a proof, refer to the supplementary material.

This concludes our formalization of creativity. We will now turn to the domain of Scratch and elaborate on how we implement our proposed creativity measure for the three modalities of code, images, and sound.

## 4 Creativity in Scratch

Scratch is a block-based programming environment with a strong focus on visual output that is used for introductory programming Maloney et al. [2010]. Fig. 2 presents a sample Scratch project, a game called “Magnetic Challenge.” In this game, the player’s task is to steer a magnet across a path of spike obstacles as fast as possible while collecting coins, using the keyboard keys or the mouse. As shown in Fig. 2, the Scratch interface is divided into five sections: In the stage area (Fig. 2(e)), the user can select a graphical background for their project. In the sprites area (Fig. 2(d)), the user can add new foreground images (sprites) to their project. Such sprites can be drawn manually, chosen from the Scratch library, or uploaded. The code blocks area (Fig. 2(a)) provides an overview of all possible code that the user may attach to sprites or backgrounds. Blocks are grouped into predefined categories, such as “motion,” “looks,” and “events.” Additionally, a user can use the extension category to load additional block packages with advanced functionality, such as “video” and “text to speech.” Finally, users can create custom blocks, which will appear in “My Blocks.”

The workspace area (Fig. 2(b)) displays all code blocks associated with the currently selected sprite or background. Users can drag and drop blocks from the code blocks area to

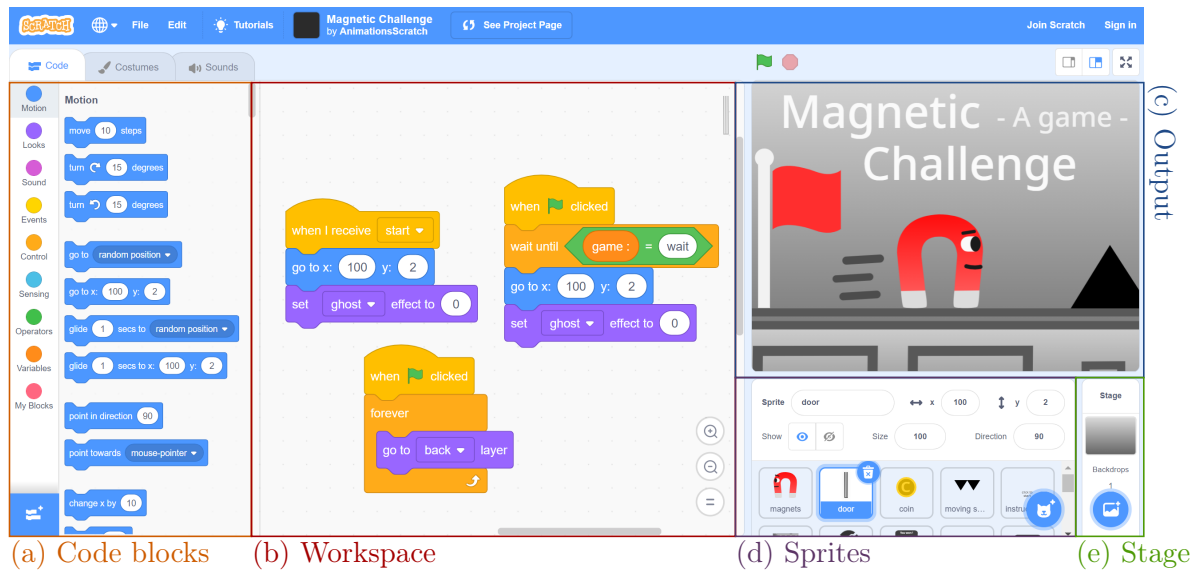


Figure 2: Screenshot of an example Scratch project. (a) Overview of possible code blocks. (b) Code blocks related to the currently selected sprite (the door). (c) Current output. (d) Overview of all sprites. (e) Stage/background.

the workspace and combine them in the workspace as needed. Each block combination starts with an event and continues with blocks that are executed sequentially whenever the event occurs. For example, Fig. 2(b) displays all blocks associated with the “door” sprite in the “Magnetic Challenge” game. Finally, the output area (Fig. 2(c)) executes the project such that the user can interact with it.

Importantly, Scratch programs are multimodal, including code, images, and sound. To fully account for the creativity in a Scratch project, we want to include all three modalities in our creativity measure. We do so by applying our framework from Section 3 to each modality separately, meaning that for code, images, and sound we obtain a measure of fluency, flexibility, and originality, respectively. The fact that our framework applies to all three modalities attests to the generality of our methods. In Section 5.4, we combine fluency, flexibility, and originality from code, images, and sound into an overall creativity measure via a machine learning model. This model is trained to match human creativity assessments.

#### 4.0.1 Code creativity

Similar to previous works, we represent a Scratch project’s code as a collection of syntax trees Price et al. [2017] one representing the stage, and one per sprite. The concept space  $\mathcal{X}$  contains all predefined blocks and extension blocks in the Scratch language, as well as user-defined custom blocks, and the null concept. We define the distance metric  $\delta$  as the shortest-path distance in the semantic network in Fig. 3. Our network models the hierarchy of Scratch blocks, where we distinguish between predefined blocks, extension blocks, and custom blocks. Within each category, we further distinguish subcategories as defined by the Scratch language. For predefined blocks, these are motion, events, control, etc. (Fig. 2(a)). For extensions, these are the single extension packages, such as video and text to speech. For custom blocks, the Scratch environment does not provide subcategories.

The edge labels in Fig. 3 represent the length of each edge. The distance between any two

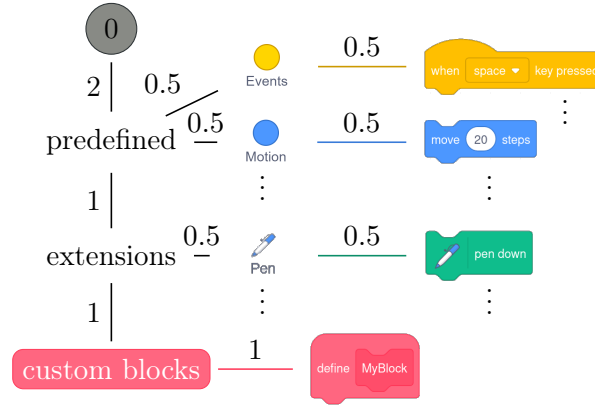


Figure 3: Semantic network, organizing the concept space  $\mathcal{X}$  and distance  $\delta$  for Scratch blocks. Code blocks are distinguished into predefined, extension, and custom blocks, with subcategories for predefined (events, motion, ...) and extensions (pen, translate, ...). The distance between blocks corresponds to the shortest path distance in the network. For example,  $\delta(\text{move}, \text{pen down}) = 0.5 + 0.5 + 1 + 0.5 + 0.5 = 3$ .

blocks equals sum over all edges that we need to traverse to get from one block to the other. For example, the distance between the “move” block and the “when key pressed” block is two due to the following calculation.

$$\begin{aligned} \delta(\text{move}, \text{when key pressed}) &= \delta(\text{move}, \text{Motion}) + \delta(\text{Motion}, \text{predefined}) + \\ &\delta(\text{predefined}, \text{Events}) + \delta(\text{Events}, \text{when key pressed}) = 0.5 \cdot 4 \end{aligned}$$

Overall, we obtain a distance of one between blocks in the same subcategory, a distance of two between blocks in different subcategories but the same category, a distance of three between predefined and extension blocks or between extension blocks and custom blocks, and a distance of four between predefined and custom blocks.

The distance to the nullconcept reflects how much programming knowledge and effort is required to use the block. For predefined blocks, we obtain a distance of three, for extension blocks a distance of four, and for custom blocks a distance of five.

Based on this distance, we computed code fluency, flexibility, and originality of 45 Scratch projects. Table 1 presents the resulting statistics.

We compute fluency via 1, which counts how many blocks are contained in a program, with a custom block being “worth” five points of fluency, an extension block four points, and a predefined block three points. For example, in the “Magnetic Challenge” program, the “go to” block presented in Fig. 2 is a predefined block from class “motion”. The program gets three points for this block, and an overall fluency score of 21 484, which is higher than the average fluency score of the projects we assessed. The high number occurs because we use squared  $\delta$  in practice, which has the added value of making fluency interpretable as a squared length, and flexibility as a variance.

For the flexibility measure, we first remove all duplicated blocks as recommended in Section 3.3. For example, the project “Magnetic Challenge” uses the “go to” block and “when flag clicked” block more than once (Fig. 2(b)). Then, we compute 2. For example, the distance between the “go to” block and the “when flag clicked” block contributes two to flexibility. Overall, after normalizing by the number of the unique blocks  $|\tilde{V}_s| = 72$  in the program, it

Table 1: Code Creativity Statistics

	fluency	flexibility	originality
mean	13 342.46	372.16	4105.02
std	19 398.40	128.60	2848.86
min	883	89.75	834
max	119 849	747.82	23 468

obtains a flexibility score of 477.92 which is higher than the average flexibility score of the projects we assessed.

Originality requires a sample of projects, meaning we need a reference point with respect to which a project is original or not. This follows prior work by Torrance [1972], Runco and Jaeger [2012]. Since the choice of sample can influence originality, we considered three reference samples, two with 20 programs, one with 10 programs. The samples were chosen randomly out of our 45 unique Scratch projects with overlap. Depending on the sample, the “Magnetic Challenge” program in Fig. 2 got three different scores for originality: 7023.05, 6641.53, and 3260.5. The average originality across projects is 4105.02 (see Table 1), indicating that the code originality of “Magnetic Challenge” is relatively high.

Further, to compute originality as defined in Section 3.3, we need an alignment distance between Scratch projects. To do so, we follow the three-step approach of Price et al. [2017] for block-based programming. First, we compute the tree edit distance Zhang and Shasha [1989] between the stage syntax trees of both programs. Then, we compute all pairwise tree edit distances between sprites in both programs. Finally, we feed this result into the Hungarian algorithm Munkres [1957] to obtain an optimal matching between the sprites. Regarding computational efficiency, we obtain a time complexity of  $\mathcal{O}(m_{\text{stage}}^2 \cdot n_{\text{stage}}^2 + m \cdot n \cdot m_{\text{sprite}}^2 \cdot n_{\text{sprite}}^2 + (m + n)^3)$  where  $m_{\text{stage}}$  and  $n_{\text{stage}}$  are the number of blocks in the stage,  $m_{\text{sprite}}$  and  $n_{\text{sprite}}$  are the maximum number of blocks per sprite, and  $m$  and  $n$  are the number of sprites. Because all these numbers are usually quite small in Scratch projects, this computation is still fast Price et al. [2017].

#### 4.0.2 Visual creativity

Projects in Scratch may contain different images, either provided by Scratch, drawn by the users, or uploaded by them. The concept space  $\mathcal{X}$  consists of all possible images that could be included in a project. Note that this space is infinitely large and varied, including images of various sizes, file formats, content, etc. Accordingly, we decide to apply a semantic embedding approach, where we embed images in a shared space before we compare them Kenett [2019], Landauer et al. [1998]. But what is an appropriate, semantic embedding for images? In computer vision, tremendous progress has been achieved via deep neural networks Feng et al. [2019], which have achieved state-of-the-art performance in challenging tasks such as object detection Lin et al. [2019], image classification Xia et al. [2017], and fault diagnosis Wen et al. [2019]. Note that all these tasks concern the semantics of an image, such that deep neural networks appear as a useful tool for our purpose. In particular, we select ResNet50 He et al. [2016]. ResNet50 is a convolutional deep neural network with residual connections, which achieved state-of-the-art results in the famous ImageNet competition He et al. [2016]. It has been shown to capture semantic differences between images in tasks such as medical image classification Reddy and Juliet [2019], software classification Rezende et al. [2017], and many more.

Table 2: Visual Creativity Statistics

	fluency	flexibility	originality
mean	102.74	27.75	0.57
std	106.53	33.69	0.03
min	4.00	0.60	0.52
max	583.00	192.00	0.66

ResNet50 translates each image into a vector. Accordingly, we represent the images in a Scratch project as a set of vectors, one per image. To compare the image vectors, we chose the Cosine distance  $\delta(\vec{x}, \vec{y}) = 1 - \frac{\vec{x}^T \cdot \vec{y}}{\|\vec{x}\| \|\vec{y}\|}$  because it is invariant against effects of scale and size. Using this distance, we computed the visual fluency, flexibility, and originality for 45 Scratch projects. Table 2 presents the resulting statistics.

Visual fluency is computed via 1, which collapses to  $\text{Flue}(s) = |V_s|$  because the Cosine distance to the zero vector is always  $\delta(x, 0) = 1 - 0 = 1$ . As stated in Section 3, this is equivalent to the fluency definition in Torrance’s test. For example, the “Magnetic Challenge” project contains  $|V_s| = 105$  images. This is higher than the average fluency score of the projects we assessed (Table 2).

To measure visual flexibility, we use 2, where  $x$  and  $y$  represent the different images in a program, and  $|V_s|$  is the total number of these images. For example, the “Magnetic Challenge” project contains a lot of images, but many of them are visually similar, such as four variants of spikes which only differ in the number of spikes and their orientation. Accordingly, the cosine distance between the corresponding ResNet50 embeddings is very low, which in turn means that they do not contribute much to flexibility. Overall, the flexibility score for “Magnetic challenge” is 23.29, which is below the average flexibility score of 27.75 (Table 2), suggesting that the visual flexibility of “Magnetic challenge” is relatively low.

We compute visual originality as the average pairwise distance between the images in the project and the images in reference projects. As with code, we used three randomly sampled groups of projects for reference. For the “Magnetic Challenge” project, we thus obtained originality scores of 0.56 in the first group, 0.55 in the second, and 0.52 in the third. The first two scores are close to the mean score of the programs in our study, and the third is the minimum score as shown in Table 2, indicating that the visual originality of “Magnetic challenge” is relatively low.

#### 4.0.3 Audio creativity

Projects in Scratch can contain different sounds, either predefined by Scratch, recorded by the user, or uploaded by the user. As with images, the space of possible sounds is infinitely large and varied, including various file formats, lengths, pitch, etc. Accordingly, we opt for a semantic embedding approach again. However, in contrast to images, we do not represent sounds as a single vector, but as a time series of vectors to take into account how sound can change over time. As a tool for our embedding, we use the pyAudioAnalysis Python package Giannakopoulos [2015]. PyAudioAnalysis has been used to extract audio features statistics which describe the wave properties of a sound, including Fourier frequencies, energy-based features, Mel-Frequency Cepstral Coefficients (MFCC), and similar representations Giannakopoulos [2015]. For each sound we extracted 136 features. As the sounds differ in sampling rates, we used different window and step sizes: 220 frames for smaller sampling rates (such as 11 025 Hz, 22 050 Hz and 24 000 Hz) and 250 for larger sampling rates (such as

Table 3: Audio Creativity Statistics

	fluency	flexibility	originality
mean	9 160	44 771 043	1744
std	12 744	69 719 132	849
min	41.20	0	1009
max	51 983	316 353 033	4379

44 100 Hz and 48 000 Hz). Each project is then represented as set of 2-dimensional matrices, each representing a sound. As a distance metric  $\delta$ , we compute the Euclidean distance between features at matching time stamps as suggested in previous sound comparisons studies San Segundo et al. [2017]. As the length of the sounds can differ, we apply padding on the end of the shorter representation before calculating the distance as suggested by Le Bel [2017]. Using this distance, we computed audio fluency, flexibility, and originality for 45 Scratch projects. Table 3 presents the resulting statistics.

Audio fluency is computed using 1, where we represent the null concept as a 2-D array filled with zeros. For example, “Magnetic Challenge” received a fluency score of 1771.69, which is much lower than the average fluency score of 9160.

For audio flexibility, we measure the distance between each pair of sounds in the program using 2.  $x$  and  $y$  are different sounds in a program and  $|V_s|$  is the total number of these sounds. The “Magnetic Challenge” program has six sounds ( $|V_s| = 6$ ), and a flexibility score of 1 177 416 which is about a quarter of the mean score of the programs’ in our study as shown in Table 3. This indicates that the audio flexibility of “Magnetic Challenge” is relatively low, as well.

Finally, audio originality is computed as the average distance of each sound in a program to sounds in a sample of reference programs. As with code and visual creativity, we used three different samples, yielding originality scores of 1126.93, 1204.83 and 1055.61 for the “Magnetic Challenge” project. All three are lower than the mean score of audio originality in our study, indicating that the audio originality of “Magnetic Challenge” is relatively low.

Overall, the example of “Magnetic Challenge” indicates that we can pinpoint which modality of a project contributes to creativity (here: the code) and which do not (here: visual and audio). This shows that, although the creativity measure of a particular project can be relatively high in one modality, this does not necessarily imply that other modalities in a student’s project also exhibit high creativity scores. Hence, it is important to examine them all.

## 5 Human Creativity Assessment

In the previous sections, we established a theory-driven formalization of creativity and applied it to Scratch programs. In this section, we describe a user study to collect expert assessments of creativity in Scratch projects. Each expert was assigned a set of preselected Scratch projects. The experts were Scratch instructors without prior knowledge in creativity theory. Our primary research question is whether we can predict human expert assessments from the automatic creativity measures in Section 4.

We asked the experts to evaluate four different aspects of each project, namely code, visuals, audio, and idea. The first three aspects capture the different modalities of Scratch projects, whereas the latter aspect refers to the general idea behind the project. As Scratch enables creating different types of projects (stories, tutorials, games, etc.) in diverse domains (math, fashion, medicine, TV shows, etc.), the idea aspect can be important to capture crucial

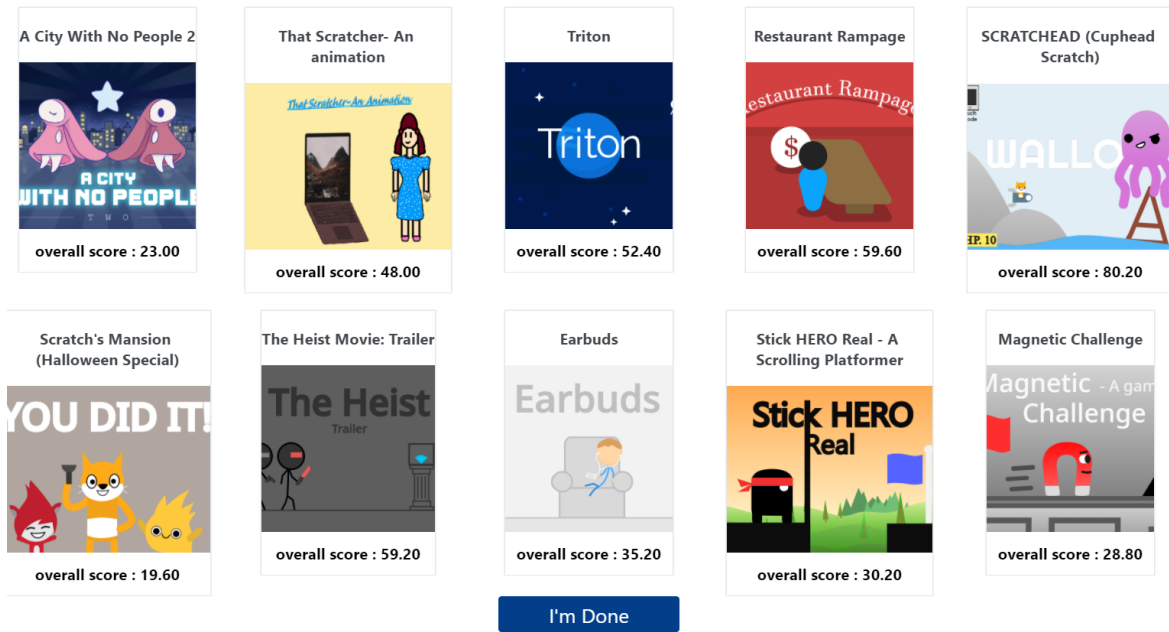


Figure 4: Home screen of the assessment tool.

context for an experts' assessment Resnick et al. [2009]. Additionally, we asked experts to weigh each aspect according to its importance for creativity. The overall creativity score for each project corresponded to the weighted sum of the creativity assessments of each aspect.

## 5.1 Scratch Creativity Assessment Tool

To facilitate the assessment process, we developed a web-application named “Scratch Creativity Assessment Tool.” Experts started on the home screen (Fig. 4), which displayed all of the projects that are assigned to an expert. The creativity score of evaluated projects also appeared, allowing experts to compare scores and change them at will.

By clicking on a thumbnail on the home screen, experts could navigate to a creativity questionnaire for the respective project. Each questionnaire began with an interactive display of the project, such that experts could test it before assessing its creativity. The questionnaire was, then, divided into five sections, namely one per aspect (code, visuals, audio, and idea) and a final section for the creativity assessment. The questions in the first four sections were designed to prime the expert's creativity assessment by presenting them with possible creativity criteria relating to fluency, flexibility, and originality in Scratch projects. For example, the questionnaire regarding the visual aspects (Fig. 5(a)) contained the binary question “Does the project contain images created by the user?”, related to fluency. If the answer was “yes,” then questions about these images were revealed, for example, “Rate the novelty of these images,” related to originality. An expert could give a rating between 0 and 100 for such questions.

Fig. 5(b) shows questions relating to the project code, such as evaluating the code complexity, efficiency, and novelty, as well as rating the effort put into the code. Questions about the project idea asked the experts to include a short description of the project's idea as they understand it, and to rate how much novelty and effort were required for developing the idea. If the project included sounds, experts were asked if these sounds were recorded by the user, uploaded, or were predefined by Scratch. Additionally, the experts rated the novelty of the sounds and the effort invested in the audio aspect.

Figure 5 displays three sections of appraisal questions used in the user study. Each section consists of a series of questions and sliders for rating different aspects of a project.

**(a) Visual Aspects -**

- Does the project contain images provided by Scratch?
- Were these images modified?
- Does the project contain imported images?
- Does the project contain images created by the user?
- Rate the novelty of these images: 0 (Familiar) to Never seen something like that
- Rate the quality of the animations: 0 (Simple) to Super realistic
- Rate the efforts put into the project's visual aspects: 0 (None) to Lots

**(b) Code Aspects -**

- Does the project contain extensions?
- Does the project contain costume blocks?
- Rate the code elegance: 57 (Spaghetti) to Modular and easy to follow
- Rate the code complexity: 0 (Simple) to Complex
- Rate the code efficiency: 0 (Wasteful) to Efficient
- Rate the novelty of the code: 0 (Familiar) to Never seen something like that
- Rate the efforts put into the project's code aspects: 57 (None) to Lots

**(c) Creativity Assessment -**

- Rate the creativity of the idea: 54 (Low) to High
- Rate the creativity of the code: 51 (Low) to High
- Rate the creativity of the visual aspects: 29 (Low) to High
- Rate the creativity of the audio aspects: 66 (Low) to High
- Describe other aspects you noticed during the assessment:
- Rate the creativity of these other aspects: 32 (Low) to High

Figure 5: Appraisal questions in the user study. (a) Questions on visual creativity. (b) Questions on code creativity. (c) Overall creativity assessment.

In the final section of the questionnaire (Fig. 5(c)), we asked experts to provide creativity scores for each aspect on a scale from 0 to 100. In order to capture aspects that went beyond code, visuals, audio, or idea, we included a text field where experts could include any additional aspect related to creativity, and an additional rating for the creativity of these additional aspects.

Once experts were done with assessing the creativity of the projects, they returned to the home screen and clicked on “I’m done” (Fig. 4). This led them to a final screen where we asked them to weigh the different aspects according to their importance for creativity. Weights were automatically scaled to values between zero and one, and normalized to a sum up to one.

This concludes our description of the Scratch creativity assessment tool. We now continue to describe our expert recruitment scheme and our study results.

## 5.2 Expert Assessment

For our study, we recruited five experts from four countries: Cuba, Vietnam, India, and Israel. All experts had at least two years experience in teaching Scratch to students of different ages in schools and after-school activities.

We selected 45 unique projects of different types (games and stories), created by different users (age between 9 and 18, from 25 different countries, with 4 to 258 created projects). The assignment of programs to the experts was randomized uniformly. Four experts evaluated 20 projects, each, while one evaluated ten projects. 80% of projects were reviewed by more than one expert. In the remainder of this paper, we designate our experts by the digits one to five.

Table 4 shows the weights assigned by the experts to the different aspects. On average, experts assigned the highest weights to idea (range .25–.3; mean .29) and visuals (range .2–.3; mean .28), a medium weight to code (range .2–.3; mean .23), a low weight to audio (range .1–.2; mean .15), and a very low weight to other aspects (range .02–.1; mean .05). This highlights that a single modality is likely insufficient to capture the full richness of creativity in Scratch projects. On the other hand, the high weight for the idea aspect is a challenge for automatic creativity assessment as the idea of a project is particularly difficult to capture in an automatic computation.



Table 4: Expert Importance Weights for each aspect

Expert	Code	Visual	Audio	Idea
1	.25	.35	.10	.25
2	.22	.28	.15	.30
3	.20	.20	.20	.30
4	.20	.30	.15	.30
5	.30	.25	.13	.30
mean	.23	.28	.15	.29

Table 5: Expert Creativity Scores

Expert	Statistic	Code	Visual	Audio	Final score
1	Mean	69.55	75.85	45.95	67.59
	SD	24.04	24.97	23.80	21.31
2	Mean	66.75	67.70	65.80	66.89
	SD	13.11	14.28	21.05	13.80
3	Mean	70.75	77.65	59.10	65.30
	SD	10.18	10.50	29.49	11.89
4	Mean	72.90	83.40	81.80	76.11
	SD	24.00	20.11	20.06	17.78
5	Mean	64.60	68.55	51.80	63.52
	SD	15.27	13.15	29.01	13.17

Table 5 presents the statistics of code, visual, audio, and final creativity scores provided by the experts. We note that not all the projects included audio; therefore, their score for this aspect is zero, which leads to the high standard deviations in this aspect across all experts. We note that the highest scores were provided by expert four and that this expert as well as expert one had the highest standard deviation for visual, code, and final creativity scores. By contrast, experts two and five gave slightly lower scores with smaller standard deviation.

Overall, we observe that experts vary both in their scores as well as in the importance of each aspect. In the following section, we analyze the agreement between experts in more detail.

### 5.3 Agreement Between Experts

The numeric agreement between experts on overlapping projects was affected by discrepancies in how experts graded the projects assigned to them, as seen in Table 5. For example, experts two, four, and five all evaluated the project “Magnetic Challenge”. Expert two gave this project a creativity score of 82 for the code aspect, while expert four gave it a score of 42, and expert five gave a score of 84. Table 5 suggests that experts five and two scored the code aspect on a similar range, whereas expert four gave higher scores with a larger variance.

We note that the differences between experts do not indicate a mistake or lack of expertise. It reflects the fact that creativity assessment is largely subjective Amabile [2018]. To partially compensate for this, we henceforth measure agreement using Kendall’s  $\tau$  Abdi [2007], which considers rank agreement rather than numeric agreement. This measure ranges from minus

one (complete disagreement between rankings) to plus one (perfect match) and is determined based on overlapping projects for each pair of experts.

Fig. 6 displays the number of overlapping projects (in brackets) as well as Kendall’s  $\tau$  for each pair of experts. We evaluated agreement separately for code (Fig 6(a)), visual (Fig 6(b)), and audio creativity (Fig 6(c)), as well as the weighted average of these three scores (Fig 6(d)).

Note that experts two and four had only one overlapping project, therefore Kendall’s  $\tau$  cannot be calculated for them. As shown in the Table, experts one and five mostly agree on audio creativity ( $\tau = .71$ ) but disagree on visual and code creativity as well as for the weighted score. These substantial differences in the rankings of projects in three out of the four scores suggests that they differ in their interpretation of creativity. For example, for the code aspect, the same project was ranked first by expert five and sixteenth by expert one. By contrast, experts two and five have a relatively high agreement for all aspects ( $\tau = .38$ ,  $\tau = .39$ , and  $\tau = .61$ , respectively). These experts have relatively similar scores (Table 5) and weights (Table 4), suggesting that they interpret creativity in similar ways. The highest agreement for visual and code creativity are between experts three and four ( $\tau = .67$  and  $\tau = .91$ ). Experts one and five have the highest agreement on the audio creativity, and experts two and five have the highest agreement on the weighted creativity ranking, although they have lower agreement on each aspect separately. This can be explained by the fact that experts two and five were more consistent in their agreement across aspects.

Despite some examples of high agreement, the average  $\tau$  in Fig. 6 is rather low ( $-.01$  for code,  $-.01$  for visual,  $.15$  for audio, and  $-.13$  for the weighted combination). We note that this low agreement occurred even though we had primed the experts with the same questionnaire, indicating that even a shared frame of reference is not necessarily sufficient to achieve consistent creativity assessments from human experts. These expert disagreements underline the challenge in quantifying creativity.

Next, we develop a machine learning model to predict human-like creativity assessments from the automatic creativity scores described in Section 4.

## 5.4 Predicting Creativity Scores

Our aim in this section is to build a machine learning model which receives the automatic creativity scores of Section 4 as input and outputs an overall creativity score for a Scratch project that is similar to a score a human expert would give. Such a model could serve as surrogate for a human expert panel and could support students and teachers in assessing the creativity of Scratch projects swiftly and frequently. We note that we can not expect a high accuracy of such a model, given that experts disagree with each other, leading to highly noisy training data. Nonetheless, we are checking whether a first step in this direction is possible, and whether we can obtain a model that agrees with each expert more than they agree with each other.

We use an XGBoost regressor Chen and Guestrin [2016] to predict the expert creativity scores for each project. As input features we used the computed originality, flexibility, and fluency measures for visual, code, and audio aspects, as described in Section 4. This provided us with nine features for each project. The reference sample  $S$  for computing originality included all other projects that the each expert rated.

We created two types of XGBoost models: First, a single-expert model trained on projects for each expert separately and, second, a combined model trained on the projects from all experts together. We note that for the combined model, projects that were evaluated by more than one expert were treated as different instances.

For each type of model, we created four different prediction models, 1) predicting the code creativity score, 2) predicting the visual creativity score, 3) predicting the audio creativity

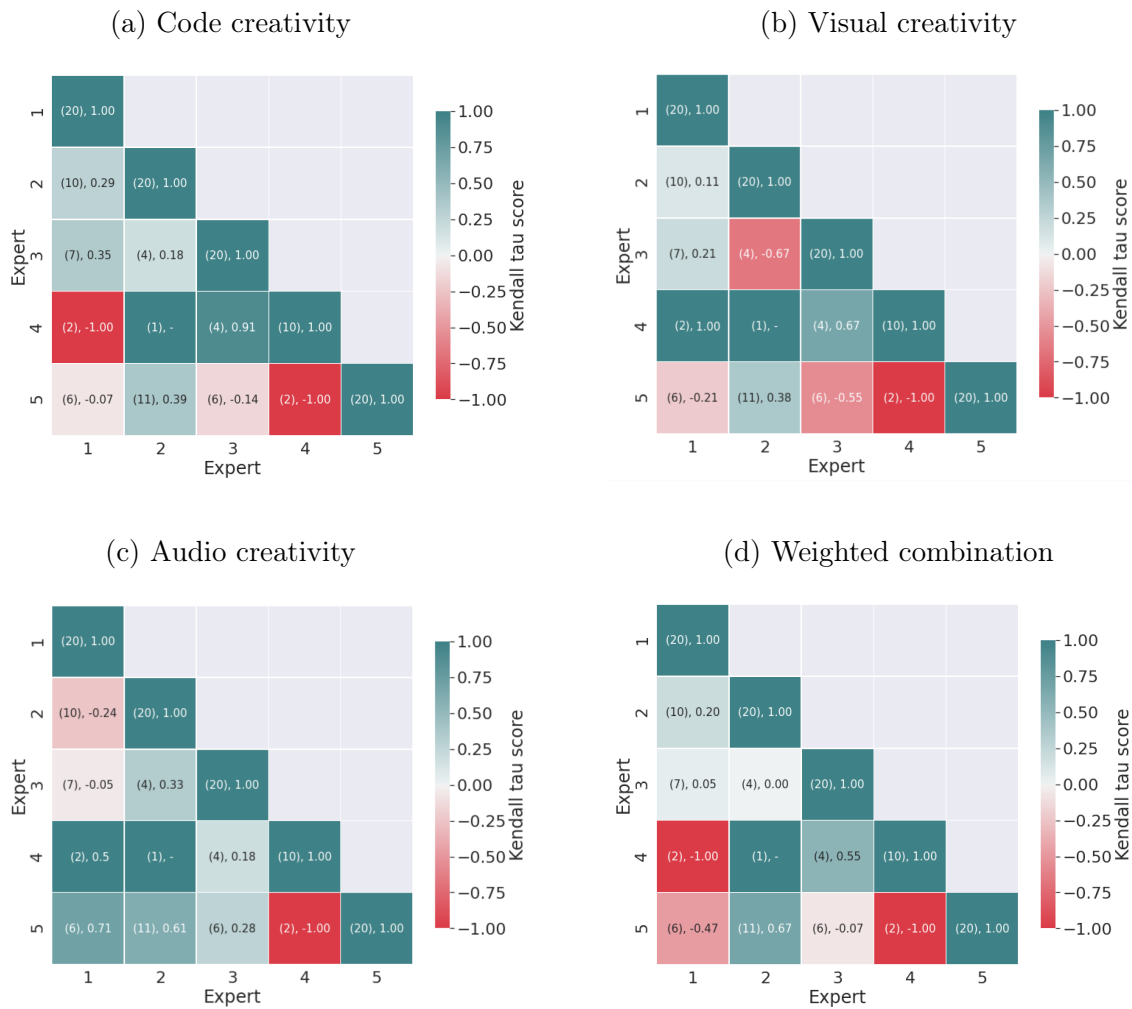


Figure 6: Kendall’s  $\tau$  between pairs of experts with overlapping projects. (The number of overlapping projects is shown in parentheses.) (a) Code creativity. (b) Visual creativity. (c) Audio creativity. (d) Weighted sum of visual, code, and audio creativity score.

Table 6: Kendall’s  $\tau$  Between XGBoost Regressor and Experts Scores

Expert	Code	Visual	Audio	Weighted combination
1	.52	.52	.49	.48
2	.51	.42	.32	.45
3	.53	.58	.69	.45
4	.46	.52	.43	.53
5	.52	.42	.44	.53
Combined	.43	.44	.41	.46

score, and 4) predicting the overall project score by the weighted combination score of visual, code, and audio. The features consisted of the originality, flexibility, and fluency for the code aspect (model 1), the originality, flexibility, and fluency for the visual aspect (model 2), the originality, flexibility, and fluency for audio aspect (model 3), or all of them (model 4), that is, all nine features (three for originality, three for flexibility, and three for fluency).

The combined model was evaluated using tenfold crossvalidation. To ensure that the test set contained at least two projects, the single-expert models were evaluated using five folds.

The combined model and the single-expert models were developed using the official implementation of XGBoost Chen and Guestrin [2016]. We selected the hyperparameters based on the structure of our data. In more detail, we set the upper complexity limit of the model to ten trees for the expert with ten projects, fifteen for the rest of the experts, and 29 trees for the combined experts. We set the maximum tree depth based on the number of features.

We evaluated the creativity score prediction via the following steps. First, we predicted creativity scores for test data projects. Second, we combined these scores with the training data scores to obtain a ranked list of all projects. Finally, we computed Kendall’s  $\tau$ , comparing this ranked list with the expert-given ranked list. This approach was applied using only the pairs with at least one instance from the test data. This method yielded sufficiently many pairs to compute Kendall’s  $\tau$ .

Table 6 presents the resulting  $\tau$  values. As seen from the table, when predicting the creativity ranking for visual and code aspects, we achieve a  $\tau$  of .51 and above for three out of five experts. This score is higher than that of the inner-agreement between the experts themselves that is reported in Fig. 6(a) and 6(b) (except for the pairs three and four, and one and four). For audio creativity, we achieve a  $\tau$  of .42 or higher for four out of five experts. This score is higher than the inner-agreement between the majority of experts pairs (except for the three pairs (1, 5), (1, 4), and (2, 5)) as shown in Fig. 6(c). When predicting the weighted creativity score, we achieve a  $\tau$  of over .45 for all experts. This score is higher than that of the inner-agreement between the majority of experts (except for two pairs (2, 5), and (3, 4)) as suggested by Fig. 6(d).

The bottom row in Table 6 presents the results for the combined XGBoost regressor model. Given the disagreement between experts, we would expect that this model performs worse than the single-expert models. Indeed, this is the case for code creativity. However, for audio creativity, the combined model achieves a higher agreement than the model for expert two, for visual creativity it achieves a higher agreement than the model for experts two and five, and for the weighted combination it achieves a higher agreement than the model for experts two and three. This indicates that the combined model can aggregate data from multiple experts in a useful way, despite the disagreement between experts.

For all aspects, the combined model achieves  $\tau$  above .41, which is higher than the agreement scores between most pairs of experts—eight out of nine for the code aspect, six out of

nine for the audio aspect, and seven pairs out of nine for the visual aspect as well as the weighed combination—and clearly higher than the average  $\tau$  between experts ( $-.13$  for the weighted combination). Overall, our results indicate that an automatic model is able to produce creativity scores which are human-like and which form a compromise between multiple experts.

## 6 Limitations

While our implementation for Scratch provides first evidence that automatic creativity scores can approximate human scores, there are still limitations of our study. First, we considered a rather small sample of experts and our results are not necessarily representative for a broader group of experts. Second, our machine learning model is trained on the expert’s data. For a proper validation of our model, our scores would have to be compared with a new group of experts. Third, our fluency model for images and code was rather coarse-grained, especially for custom code blocks. Future work could improve this model by including more nuanced measures, such as code complexity Moreno-León et al. [2016]. Fourth, we used a random reference sample for creativity. However, we could inject more domain knowledge by adjusting our reference sample, for example by selecting the demo projects supplied by Scratch or the standard image set provided by Scratch. Fifth, our current scheme is a static assessment. Future work could investigate how students’ creativity scores evolve over time and how teaching impacts creativity scores. Finally, any scheme for (automated) assessment raises ethical issues, such as fairness, and ought to be implemented with care. Future work should investigate how creativity assessment impacts students, for example, with respect to their confidence and self-image.

## 7 Conclusion

We introduced a novel measure of creativity with three components: the distance to an empty product (fluency), the distance between concepts in the product (flexibility), and the distance to typical products (originality). Our measure only requires two ingredients: a set of concepts that students can use and a (semantic) distance between those concepts. This makes our measure easily applicable across modalities. For example, in this work, we represented code by its syntactic building blocks, we represented images by neural network embeddings, and we represented sound as a sequence of audio features. We showed that our proposed measure is a proper generalization of fluency, flexibility, and originality as proposed by Torrance [1972].

To validate our distance-based creativity measure, we applied it to Scratch projects and compared it to the creativity assessment of human experts. We found that an XGBoost model using fluency, flexibility, and originality as input agreed at least as well with human assessments as humans agreed with each other. This is partially because human experts tended to disagree, indicating again that creativity is hard to quantify. But it also provides some evidence that automatic measures of creativity can approximate human assessments of creativity.

## Acknowledgment

This work was partly funded by the German Research Foundation under Grant PA 3460/2-1

## References

- Hervé Abdi. The Kendall rank correlation coefficient. In Neil Salkind, editor, *Encyclopedia of Measurement and Statistics*, volume 1, pages 508–510. Sage, Thousand Oaks, CA, USA, 2007.
- Teresa M Amabile. *Creativity in Context: Update to the Social Psychology of Creativity*. Routledge, Milton Park, UK, 2018.
- Philip Bille. A survey on tree edit distance and related problems. *Theor. Comput. Sci.*, 337(1):217–239, June 2005. doi:10.1016/j.tcs.2004.12.030.
- Margaret A Boden. *The Creative Mind: Myths and Mechanisms*. Routledge, Milton Park, UK, 2004.
- Sebastien Bogleux, Luc Brun, Vincenzo Carletti, Pasquale Foggia, Benoit Gaüzère, and Mario Vento. Graph edit distance as a quadratic assignment problem. *Pattern Recogn. Lett.*, 87:38–46, February 2017. doi:10.1016/j.patrec.2016.10.001.
- Jon Bustillo and Pablo Garaizar. Using Scratch to foster creativity behind bars: Two positive experiences in jail. *Thinking Skills and Creativity*, 19:60–72, 2016. doi:10.1016/j.tsc.2015.08.003.
- Nick Chater and Ulrike Hahn. Representational distortion, similarity and the universal law of generalization. In *Proc. Interdisciplinary Workshop Similarity and Categorization (SimCat '97)*, pages 31–36, Edinburgh, UK, November 28–30, 1997.
- Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery and Data Mining (KDD '16)*, pages 785–794, San Francisco, CA, USA, August 13–17, 2016. doi:10.1145/2939672.2939785.
- Xin Feng, Youni Jiang, Xuejiao Yang, Ming Du, and Xin Li. Computer vision algorithms and hardware implementations: A survey. *Integration*, 69:309–320, November 2019. doi:10.1016/j.vlsi.2019.07.005.
- Georgi V. Georgiev and Danko D. Georgiev. Enhancing user creativity: Semantic measures for idea generation. *Knowl.-Based Syst.*, 151:1–15, 2018. doi:10.1016/j.knosys.2018.03.016.
- Theodoros Giannakopoulos. pyaudioanalysis: An open-source Python library for audio signal analysis. *PLoS ONE*, 10(12), December 2015. doi:10.1371/journal.pone.0144610.
- Michail N. Giannakos, Letizia Jaccheri, and Roberta Proto. Teaching computer science to young children through creativity: Lessons learned from the case of Norway. In *Proc. 3rd Comput. Sci. Educ. Res. Conf. (CSERC '13)*, pages 103–111, Arnhem, Netherlands, April 4–5, 2013. URL <https://dl.acm.org/doi/abs/10.5555/2541917.2541927>.
- Joy Paul Guilford. The structure of intellect. *Psychol. Bull.*, 53(4):267–293, July 1956. doi:10.1037/h0040755.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Comp. Vision and Pattern Recognit. (CVPR '16)*, Las Vegas, NV, USA, June 27–30, 2016. doi:10.1109/CVPR.2016.90.

- Danah Henriksen, Punya Mishra, and Petra Fisser. Infusing creativity and technology in 21st century education: A systemic view for change. *Educ. Technol. & Soc.*, 19(3):27–37, July 2016. URL <https://www.jstor.org/stable/10.2307/jeductechsoci.19.3.27>.
- Arnon HersHKovitz, Raquel Sitman, Rotem Israel-Fishelson, Andoni Egufluz, Pablo Garaizar, and Mariluz Guenaga. Creativity in the acquisition of computational thinking. *Interact. Learn. Environ.*, 27(5-6):628–644, April 2019. doi:10.1080/10494820.2019.1610451.
- Carl J. Hodgetts, Ulrike Hahn, and Nick Chater. Transformation and alignment in similarity. *Cognition*, 113(1):62–79, October 2009. doi:10.1016/j.cognition.2009.07.010.
- Chun-Chieh Huang, Ting-Kuang Yeh, Tsai-Yen Li, and Chun-Yen Chang. The idea storming cube: Evaluating the effects of using game and computer agent to support divergent thinking. *Educ. Technol. & Soc.*, 13(4):180–191, October 2010. URL <https://www.jstor.org/stable/10.2307/jeductechsoci.13.4.180>.
- Rotem Israel-Fishelson, Arnon HersHKovitz, Andoni Egufluz, Pablo Garaizar, and Mariluz Guenaga. The associations between computational thinking and creativity: The role of personal characteristics. *J. Educ. Comput. Res.*, 58(8):1415–1447, January 2021. doi:10.1177/0735633120940954.
- Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *Proc. IEEE Conf. Comp. Vision and Pattern Recognit. (CVPR '15)*, Boston, MA, USA, June 7–12, 2015. doi:10.1109/CVPR.2015.7298990.
- Yoed Kenett. What can quantitative measures of semantic distance tell us about creativity? *Current Opinion Behav. Sci.*, 27:11–16, 2019. doi:10.1016/j.cobeha.2018.08.010.
- Kyung Hee Kim. Can we trust creativity tests? A review of the Torrance tests of creative thinking (TTCT). *Creativity Res. J.*, 18(1):3–14, 2006. doi:10.1207/s15326934crj1801\_2.
- Maria Knobelsdorf and Ralf Romeike. Creativity as a pathway to computer science. In *Proc. 13th Annu. Conf. Innov. and Technol. in Comput. Sci. Educ. (ITiCSE '08)*, pages 286–290, Madrid, Spain, June 30–Jul. 2, 2008. doi:10.1145/1384271.1384347.
- Anastasia Kovalkov, Avi Segal, and Kobi Gal. Inferring creativity in visual programming environments. In *Proc. 7th ACM Conf. Learn. at Scale (L@S '20)*, pages 269–272, virtual event, August 12–14, 2020. doi:10.1145/3386527.3406725.
- Anastasia Kovalkov, Benjamin Paaßen, Avi Segal, Kobi Gal, and Niels Pinkwart. Modeling creativity in visual programming: From theory to practice. In *Proc. 14th Int. Conf. Educ. Data Mining (EDM '21)*, virtual event, June 29–Jul. 2, 2021. URL [https://educationaldatamining.org/EDM2021/virtual/static/pdf/EDM21\\_paper\\_44.pdf](https://educationaldatamining.org/EDM2021/virtual/static/pdf/EDM21_paper_44.pdf).
- Thomas K Landauer, Peter W Foltz, and Darrell Laham. An introduction to latent semantic analysis. *Discourse Processes*, 25(2-3):259–284, 1998. doi:10.1080/01638539809545028.
- Frédéric Le Bel. Agglomerative clustering for audio classification using low-level descriptors. Technical report, Pédagogie et action culturelle, IRCAM, Paris, France, March 2017. URL <https://hal.archives-ouvertes.fr/hal-01491270/>.
- Vladimir Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Sov. Phys. Dokl.*, 10(8):707–710, February 1966.

- Chunmian Lin, Lin Li, Wenting Luo, Kelvin CP Wang, and Jiangang Guo. Transfer learning based traffic sign recognition using Inception-v3 model. *Periodica Polytechnica Transp. Eng.*, 47(3):242–250, May 2019. doi:10.3311/PPtr.11480.
- John Maloney, Mitchel Resnick, Natalie Rusk, Brian Silverman, and Evelyn Eastmond. The Scratch programming language and environment. *ACM Trans. Comput. Educ.*, 10(4), November 2010. doi:10.1145/1868358.1868363.
- A.B. Markman and D. Gentner. Structural alignment during similarity comparisons. *Cogn. Psychol.*, 25(4):431–467, October 1993. doi:10.1006/cogp.1993.1011.
- Bassam Mokbel, Sebastian Gross, Benjamin Paaßen, Niels Pinkwart, and Barbara Hammer. Domain-independent proximity measures in intelligent tutoring systems. In *Proc. 6th Int. Conf. Educ. Data Mining (EDM '13)*, pages 334–335, Memphis, TN, USA, July 6–9, 2013. URL [https://www.educationaldatamining.org/EDM2013/papers/rn\\_paper\\_68.pdf](https://www.educationaldatamining.org/EDM2013/papers/rn_paper_68.pdf).
- Jesús Moreno-León, Gregorio Robles, and Marcos Román-González. Comparing computational thinking development assessment scores with software complexity metrics. In *Prof. IEEE Global Eng. Educ. Conf. (EDUCON '16)*, pages 1040–1045, Abu Dhabi, UAE, April 10–13, 2016. doi:10.1109/EDUCON.2016.7474681.
- James Munkres. Algorithms for the assignment and transportation problems. *J. Soc. Ind. Appl. Math.*, 5(1):32–38, March 1957. doi:10.1137/0105003.
- Benjamin Paaßen, Barbara Hammer, Thomas Price, Tiffany Barnes, Sebastian Gross, and Niels Pinkwart. The continuous hint factory—providing hints in vast and sparsely populated edit distance spaces. *J. Educ. Data Mining*, 10(1):1–35, June 2018. URL <https://jedm.educationaldatamining.org/index.php/JEDM/article/view/158>.
- Elżbieta Pękalska and Robert P. W. Duin. *The Dissimilarity Representation for Pattern Recognition: Foundations and Applications*. World Scientific, Singapore, 2005.
- Thomas W. Price, Rui Zhi, and Tiffany Barnes. Evaluation of a data-driven feedback algorithm for open-ended programming. In *Proc. 10th Int. Conf. Educ. Data Mining (EDM '17)*, pages 192–197, Wuhan, China, June 25–28, 2017. URL [https://educationaldatamining.org/EDM2017/proc\\_files/papers/paper\\_36.pdf](https://educationaldatamining.org/EDM2017/proc_files/papers/paper_36.pdf).
- A. S. B. Reddy and D. S. Juliet. Transfer learning with ResNet-50 for malaria cell-image classification. In *Proc. Int. Conf. Commun. and Signal Process. (ICCSP '19)*, pages 945–949, Chennai, India, April 4–6, 2019. doi:10.1109/ICCSP.2019.8697909.
- Mitchel Resnick, Karen Brennan, Cristóbal Cobo, and Philipp Schmidt. Creative learning @ scale. In *Proc. 4th ACM Conf. Learn. at Scale (L@S '17)*, pages 99–100, Cambridge, MA, USA, April 20–21, 2017. doi:10.1145/3051457.3054034.
- Mitchel Resnick et al. Scratch: Programming for all. *Commun. ACM*, 52(11):60–67, November 2009. doi:10.1145/1592761.1592779.
- E. Rezende, G. Ruppert, T. Carvalho, F. Ramos, and P. de Geus. Malicious software classification using transfer learning of ResNet-50 deep neural network. In *Proc. 16th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA '17)*, pages 1011–1014, Cancun, Mexico, December 18–21, 2017. doi:10.1109/ICMLA.2017.00-19.



- Kelly Rivers and Kenneth R. Koedinger. Data-driven hint generation in vast solution spaces: a self-improving Python programming tutor. *Int. J. Artif. Int. Educ.*, 27(1):37–64, March 2017. doi:10.1007/s40593-015-0070-z.
- Mark A. Runco and Garrett J. Jaeger. The standard definition of creativity. *Creativity Res. J.*, 24(1):92–96, February 2012. doi:10.1080/10400419.2012.650092.
- Mark A. Runco, Ahmed M. Abdulla, Sue Hyeon Paek, Fatima A. Al-Jasim, and Hanadi N. Alsuwaidi. Which test of divergent thinking is best? *Creativity. Theories – Res. – Appl.*, 3(1):4–18, June 2016. doi:10.1515/ctra-2016-0001.
- Eugenia San Segundo, Athanasios Tsanas, and Pedro Gómez-Vilda. Euclidean distances as measures of speaker similarity including identical twin pairs: A forensic investigation using source and filter voice characteristics. *Forensic Sci. Int.*, 270:25–38, January 2017. doi:10.1016/j.forsciint.2016.11.020.
- John F Sowa. Semantic networks. In L. Nadel, editor, *Encyclopedia of Cognitive Science*. Wiley, Hoboken, NJ, USA, 2006. doi:10.1002/0470018860.s00065.
- David Spendlove. Creativity in education: A review. *Design and Technology Education*, 10(2):9–18, May 2008. URL [https://ojs.lboro.ac.uk/DATE/article/view/Journal\\_10\\_2\\_RES1](https://ojs.lboro.ac.uk/DATE/article/view/Journal_10_2_RES1).
- E Paul Torrance. Predictive validity of the Torrance tests of creative thinking. *J. Creative Behav.*, 6(4):236–252, December 1972. doi:10.1002/j.2162-6057.1972.tb00936.x.
- Long Wen, X Li, Xinyu Li, and Liang Gao. A new transfer learning based on VGG-19 network for fault diagnosis. In *Proc. IEEE 23rd Int. Conf. Comp. Supported Cooperative Work in Design (CSCWD '19)*, pages 205–209, Porto, Portugal, May 6–8, 2019. doi:10.1109/CSCWD.2019.8791884.
- Frank E Williams. Assessing creativity across Williams’ ‘cube’ model. *Gifted Child Quart.*, 23(4):748–756, December 1979. doi:10.1177/001698627902300406.
- Xiaoling Xia, Cui Xu, and Bing Nan. Inception-v3 for flower classification. In *Proc. 2nd Int. Conf. Image, Vision and Comp. (ICIVC '17)*, pages 783–787, Chengdu, China, June 2–4, 2017. doi:10.1109/ICIVC.2017.7984661.
- Yu-chu Yeh and Chun Fu Lin. Aptitude-treatment interactions during creativity training in e-learning: How meaning-making, self-regulation, and knowledge management influence creativity. *Educ. Technol. & Soc.*, 18(1):119–131, January 2015. URL <https://www.jstor.org/stable/10.2307/jeductechsoci.18.1.119>.
- Kaizhong Zhang and Dennis Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18(6):1245–1262, December 1989. doi:10.1137/0218082.