**Encoding Times Reveal The Deep Structure of Instructions**

Kento Kitano and Andrea Stocco,

University of Washington,

Seattle, WA 98195

Learning and Instructions (maybe, focused on school instructions)

Brain and Cognition (maybe)

Cognition and Instruction (maybe, focused on school instructions)

Frontiers in Psychology

Memory and Cognition

Acta Psychologica

Quarterly J of Exp Psychology

**Abstract**

Humans have the unique ability of learning new tasks from instructions, which makes them capable of quickly communicating and acquiring new skills. Recently, this ability has been formally studied under controlled laboratory conditions under the name of Rapid Instructed Task Learning (RITL). Several aspects of RITL are currently known, including the neural bases of skill acquisition and the likely neural circuits where instructions are represented. However, the specific format by which instructions are encoded by learners is not fully understood yet. In this study, we examine this issue by comparing two alternative models of instructions representation, which make different predictions on the time it takes to memorize instructions whose structure is made of different segments or “chunks”. . For this study, students at University of Washington participated in a computer-run experiments solved a variety of instructed RITL tasks based on simple arithmetical operators. It was predicted that instructions with less number of chunks result in the faster encoding instructions. Analysis showed that, across a variety of tasks, the number of chunks had a significant effect on the speed of encoding. However, the same factor had no effect on the time it took participants to execute the instructions, suggesting that efficient encoding might save time during execution. The implication of these findings for effective learning in instructions are discussed.

**1. Introduction**

The ability to learn new tasks from instructions is one of the most distinctive features of human cognition and one of the foundations of modern societies, where complex skills need to be communicated and acquired quickly and precisely. In the scientific literature, this ability is specifically called Rapid Instructed Task Learning (RITL: Cole, Laurent & Stocco, 2013), and has been recently at the center of many investigative efforts (Cole et al., 2010; Ruge & Wolfensteller, 2010; Stocco et al, 2012). Unlike traditional reinforcement learning, which depends “shaping” behavior through “trial and error” approaches based on timely feedback from the environment, RITL is described as a “first-trial” approach, in which people receive a full set of instructions for a novel task at the beginning of each trial. In other words, RITL permits humans to avoid exploring multiple routes toward the goal, instead enabling them to follow the fastest path with the highest accuracy already shown by the instructor. Because of this, successful RITL processes can be used to change behaviors faster than traditional learning methods. The advantages of RITL are apparent and ubiquitous in in daily situations. For example, driving a vehicle will be extremely difficult and dangerous without RITL, because learning to appropriately respond to traffic lights with trial and error would results in catastrophic consequences. Also, individual differences in RITL ability would might translate in the performance of students in classrooms (Stocco & Prat, 2014). Thus, a better understanding of the RITL mechanism has deep implications for educators, trainers, and private companies.

The majority of RITL studies so far have been neuroimaging studies examining changes in brain activity that occur when instructions are used to guide behavior; in RITL terms, this process is known as “instruction execution” (Stocco et al., 2012). Comparatively little is known, however, on the processes by which instructions are read and a mental representation of their meaning is formed---a phase called “instruction encoding”. Based on previous published studies (for example, Duncan et al., 1995), Cole et al (2013) have argued that instructions are represented in a non-linguistic format. Using a machine learning technique known as Multi-Voxel Pattern Analysis (MVPA: ref) , Cole et al (2013) were able to show that similar instructions are associated to similar patterns of activity in the lateral prefrontal cortex, thus suggesting that this region contains a neural representation of the instructions to be performed. Furthermore, Reverberi et al (2011) were able to show that this code is compositional; they proved that entirely new instructions can be identified by the patterns of activity elicited by their components.

Despite providing tremendous insight into the neural substrates of how instructions are represented during the process of learning, these studies have not yet answered the important question of the nature, and structure, of such representation. While many authors seem to agree that the internal representation of instructions consists of a structured template for behavior (Stocco et al 2012; Cole et al 2010; Brass et al…), alternative implementations are possible. Perhaps the most obvious alternative is that rules are represented as a list of “tokens”, each of which represents one specific step of the instruction. According to this view, when memorizing rules for a trial, a subject is simply committing to working memory a list of items that are uniquely associated to the operations performed later. In most cases, these items are probably words. For instance, because instructions in the studies by Cole et al (2010) and Stocco et al. (2012) consisted of lists of three words (e.g., “SAME/SWEET/RED”, or “ADD/TRIPLE/HALF”), participants are likely to simply commit these words to memory, and go through the list as they execute the corresponding instruction steps. In principle, the tokens committed to working memory could be any sort cue of that is associated with the instruction, even non-verbal cues such the colors and shapes used in the experiment by Ruge and Wolfensteller (2010).

In contrast to this view, instructions might be represented not as a list of steps, but as structured mental plans that contain specific information about the nature of operations to be performed. These mental plans would contain, for example, specific details on how the different operations to perform are related to each other and to the stimuli. As an example, let us consider the arithmetic RITL paradigm used by Stocco and colleagues (Stocco et al., 2012; Stocco & Prat, 2014; Becker, Prat, & Stocco, 2016). In this paradigm, instructions consist of three arithmetic operations, which determine the rules of the instruction, and the stimuli consist of two integer numbers, X and Y. The operations are to be performed consecutively, with the results of each operation being the input of the following one. All the instructions contain one “binary” operation, which must be applied to both numbers (such as “Add”, “Multiply”, or “Subtract”), and two “unary” operations, which are applied only to one number (such as “Double”, “Triple”, “Half”). Additionally, the binary operation is always applied first, and the two unary operations are always applied afterwards. Thus, a task like “ADD/TRIPLE/HALF” would be interpreted as “*Add* X and Y, then *Triple* the result, and finally *Divide* the new value by 2”, or, in mathematical notation, (((X + Y) \* 3) / 2). According to the “list” interpretation of the results, these rules of instructions are simply memorized as an ordered list of three tokens, “Add”, “Triple” and “Half”. On the other hand, according to the “Structure” view, these rules are represented in a way that details their relationship with the stimuli and with each other. For example, a proper representation would connect the input numbers X and Y with the “Add”, but not with “Triple” or “Half”, because these numbers are only the input of the first operations. This representation is more detailed and complex than a simple list, as it explicitly details how inputs and operations are connected to each other.

Note that these two alternative cases are indistinguishable on the bases of current neurological and neuroimaging evidence. The main point made by Cole et al. (2013) and Reverberi et al (2011) is that instructions are represented in a compositional fashion at the level of the neural code. Both the “list” and the “structure” view of instruction representation are compositional; the two views differ only in the richness of their contents. However, as the next section will show, the two views make different predictions about the time it takes to memorize instructions in “list” or “structure” format., and can thus be distinguished on the bases of behavioral data.

**Distinguishing Between the List and the Structure Models of Instructions**

In order to understand the nature of rule representation, we will exploit the well know phenomenon of *chunking*. Put it simply, “chunking” reflects the phenomenon that people spontaneously divide a list of items to memorize in groups or “chunks”. The grouping of items into chunks and the number of chunks might reflect any superficial or deep characteristic of the list. **(references here).**

Importantly, the way participants divide up the list of items into chunks might offer insight into the relationships between the items themselves. Consider, for example, the task of memorizing the list of letters “F B I C I A”. In principle, these items could be memorized as a set of two three-letter chunks (F B I) (C I A), or a set of three two-letter chunks (F B) (I C) (I A). Because both FBI and CIA are common and well known acronyms, however, the greatest majority of participants would prefer the former format to the latter. Conversely, if participants show a consistent preference for using the (F B I) (C I A) format, one could infer something about the relationship among items--in this case, that FBI and CIA are more familiar acronyms that FB, IC, or IA.

In this experiment, we manipulated the internal structure of instructions of different length, and recorded the time it took participants to memorize them. Specifically, the structure of instructions was manipulated as to break the instructions into two or three chunks, depending on the position of the binary operations relative to unary ones. If the memorization of instructions reflects their structure, then we should find that participants’ memorization patterns reflect the number of chunks within the structure. If, on the other hand, participant simply memorize the instructions as a list of words, then their particular chunking strategy should not be affected by the deep structure of the instructions.

To implement this paradigm, we created a modified version of the arithmetic RITL task used by Stocco et al (2012), Stocco and Prat (2014), and Becker, Prat, & Stocco (2016) and described above. In this version, participants learn instructions of varying length, from two (e.g., “ADD / DOUBLE”) to four operations (e.g., “TRIPLE / HALF / ADD / DOUBLE” ). The list of all operators and their descriptions are shown in **Table 1.** As in previous studies, all instructions had one binary operator (e.g., “ADD”) and one to three unary operators (e.g., “DOUBLE”). Different structures were created by modifying the position of the binary operator within the list. To understand how the position of the binary operator alters the structure of the instructions, consider

:The set of used in the current RITL experiment.

|  |  |  |
| --- | --- | --- |
| Name of Operator | Action of Operator | Type of Operator |
| ADD  SUBTRACT  TIMES  DIVIDE  INCREASE  DECREASE  HALF  DOUBLE  THIRD  TRIPLE | X + Y  X - Y  X \* Y  X / Y  X + 1  X - 1  X / 2  X \* 2  X / 3  X \* 3 | B (Binary)  B  B  B  U (Unary)  U  U  U  U  U |



Figure 1: A comparison of the list-based and chunk-based structures that can be used for instruction encoding. In the list-based model, instructions are simply encoded as a list of operations to perform. In the chunk-based model, a deeper format is used, whereby instructions are divided according to the position of the binary operation within the list of operations. This grouping is a consequence of the fact that the binary operation represents the entry point for the input number *Y*. As a result, the list model is only sensitive to the number of operations within instructions, while the chunk model is sensitive to both the number of operations and the number of chunks.

The binary operation plays a crucial rule to characterize the instructions because it is the only one that combines two inputs. Consecutive unary operations can be chained in sequence, with the input of each operation being the output of the previous. The binary operator, however, is peculiar because it requires two inputs, X and Y, the latter of which is only used at this very point. As a consequence, all the unary operations before it apply to the input number X and its modifications, and all the one following it apply to the result of the binary operation.

It follows that the moment the binary operation divides the flow of operations in three parts. Thus, if participants are encoding the deep structure of operations, then their natural way of “chunking” instructions should be as follows: before the binary operation, the binary operation, and after the binary operation. As Figure1 shows, this chunking structure is simplified when the binary operator is at the end or at the beginning of the instructions. In such cases, the number of chunks decreases from three to two, because either the chunk before or after the binary operation is missing.

The two types of structures (list-based and chunk-based) make different predictions about the encoding times for instructions of different kinds (Figure 1).

Here, if participants use the deep structure of instructions to create the representation of task, we first predict that chunking effect is not present in the execution time but in the encoding time. It is because if participants encode representation of task, difficulties among instructions will be minimized; in contrast, if the abstraction of instruction is incomplete or participants do not create a representation at all, the execution time would simply reflect the nature of the original instruction. Then chunk effect would be present in execution time. Given the insights from past RITL studies, our prediction for the execution phase is that execution time is solely influenced by the length of instructions. Secondly, we hypothesizes that if participants create representation from instruction, the representation takes the “structured” format instead of the “list” format. Thus encoding time would be influenced by the deep structure of instruction. Specifically, if participants indeed use the deep structure of operations to memorize instructions, then we predict that, within instructions of the same length, instructions that can be chunked in two parts (i.e., binary operation at the beginning or the end) will be encoded faster than instructions that can be chunked in three parts (binary operations in the middle).

**Method**

**Participants**

The total of twenty eight undergraduate students at University of Washington (14 males, 13 females and 1 who preferred not to disclose its gender, mean age of 21.6) participated in this study in exchange for academic credits..

**Materials**

The task used in our experiment was a modified version of the arithmetic Rapid Instructed Task Learning (RITL) paradigms used by by Stocco et al (2012; Stocco and Prat, 2014; Becker, Prat, & Stocco, 2016). This tasks consists of applying mathematical operations to pairs of numbers, with both the instructions for which operations to apply and the pairs of numbers changing at each trial. As show in Figure 2, each trial consisted of an encoding and an execution phase, each introduced by a characteristic fixation.



Figure 2, A sample trial from the modified RITL paradigm used in the experiment.

In the encoding phase, participants were shown new instructions in the form of a list of 2~4 mathematical operators on computer screen. They were asked to memorize them and press any key on a keyboard when they were sure to have the instructions memorized. At the beginning of the execution phase, participants were given two values for X and Y. Both values were integer numbers between 1 and 9. During this phase, participants had to mentally apply the list of operations memorized during the encoding phase to the two numbers. The instructions for each trial were designed in such a way that the result was always an integer number between 1 and 9. In addition, all of the intermediate results of the operations were constrained to be integer numbers between 1 and 40. Once participants had solved the problem, they typed in their answer by pressing the corresponding number key on the top row of a standard keyboard. Participants were not given any feedback on whether they solved a problem correctly, and the next trial began immediately after their response. Before starting the experiment, participants were instructed to solve the tasks as quickly and as accurately as possible. Although both encoding and execution phases were self-paced, they were both set to time-out after 20 seconds. After a time-out, the software moved to the next trial, counting the time-out trial as an error.

Altogether, participants performed 98 trials, divided into 8 blocks of 14 trials each. Each block contained 14 trials, two for each of the seven possible combinations of instructions length and structure, as depicted in Figure 1, The distribution of the 10 different operators (Table 2) across the various trial types was optimized by means of Monte Carlo simulation, ensuring that each operation had roughly the same probability of appearing in each position of each trial type.. Before the beginning of the experiment, participants performed a practice block consisting of 8 trials, . After completing each block, participants received a feedback message telling them the percentage of correct responses in the previous block.

**Procedure**

Participants performed the experiment in small groups of 1 to 4, during 1-hour sessions in an dedicated experimental room . The room was equipped with four individual testing stations, each of which was equipped with a personal computer, a standard keyboard, a standard mouse, and a 21’ LCD monitor with a 16:9 aspect ratio. The individual testing stations were placed in line and separated by opaque dividers, to prevent participants from seeing each other. At the beginning of each experiment, the experimenter randomly assigned one of the computers for each participant. As soon as they were arranged, participants signed an informed consent form and a demographic survey. Upon completing the survey, participants began the experiment on the computer. The experimenter remained in the room throughout the session..

**Results**

Overall, participants were remarkably accurate, with an average accuracy of 83.1% across all trials. Because accuracy scores are bounded and intrinsically not normally distributed, they were subjected to arcsin-root transformation before statistical analysis. As expected, a one-way ANOVA showed that accuracy was significantly affected by the length of instructions (F(2, 52) = 38.76, p < 0.0001), with 2-operation rules being easier to perform (92.3% accuracy) than 3-operation (88.3%) or 4-operation trials (74.6%). None of the other factors had any significant effect on accuracies (*F*(1, 26) < 0.70, *p* > 0.41).

Reaction times were analyzed for correct trials only. To analyze encoding times, we first eliminated all the response times that took longer than 10.878 seconds, corresponding to the upper 2.5% of the latency distribution. These trials likely reflected very different mental strategies, such as participants re-encoding instructions after a mistake. Imposing a hard cutoff also maintains the overall properties of the response times distribution while significantly reducing the effect of outliers (Ratcliff, 1993). Because the distribution of the encoding times showed remarkable skewness (skewness = 1.54) and significant non-normality ( Shapiro-Wilk, W = 0.90, *p* < 0.00001), estimates of encoding times for each type of instruction were calculated using the median of the response times for each participant, instead of the mean (Ratcliff, 1993).

The overall pattern of results is illustrated in Figure 2, which shows the encoding time for each of the nine different structures, divided according to the three lengths (2, 3, and 4 operations) and the position of the binary operation within the instructions. Not surprisingly, instruction length has a large and significant effect on encoding times (F(2, 52) = 136.20, p < 0.0001), with encoding times increasing with the number of operations.

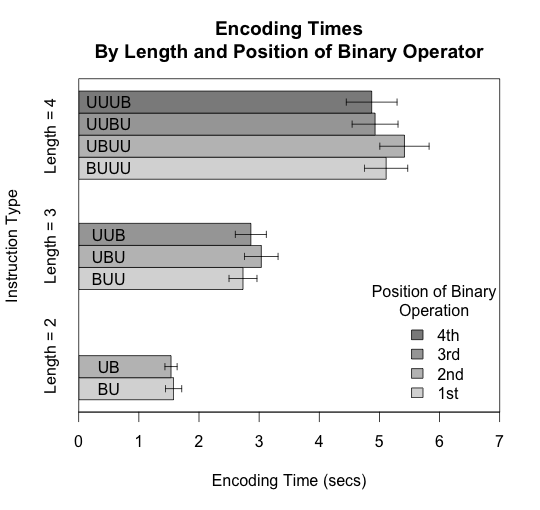


Figure 3. Mean encoding times (+/- SEM) for the seven types of instructions examined in this study. Instructions are divided by length (2, 3, or 4 operators) and by the position of the binary operator within the sequence (1st, 2nd, 3rd, or 4th). The corresponding instruction structure (e.g., “UUUB” for instructions of length 4, with binary operation in 4th position; See Figure 1) is indicated on the corresponding bar.

To estimate the separate effects of the number of chunks and the number of operators, the two-operation instructions were excluded from the analysis, and the remaining data was then aggregated by instruction length and number of chunks. This exclusion of the two-operation instructions was necessary because they always consists of two chunks. A 2x2 ANOVA, using length (3 vs. 4 operations) and number of chunks (2 vs. 3 chunks) as factors, encoding time as the dependent variable, and participants as the random factor was then carried out over the aggregated data. The analysis uncovered both a significant effect of the number of chunks (*F*(1, 26) = 4.59, *p* = 0.04) and a significant effect of instruction length (*F*(1, 26) = 155.7, *p* < 0.00001), with no interaction (*F*(1,26) = 0.09, *p* = 0.77). As predicted, instructions made of two chunks took significantly less time than instructions of three chunks when the total number of operations was three (*M* = 2.79s +/- 1.22 vs. *M* = 3.03 +/- 1.42) and when it was four (*M* = 4.99s, *SD* = 1.87 vs. *M* = 5.17, *SD* = 1.85). Figure 4 provides a visual summary of these findings.

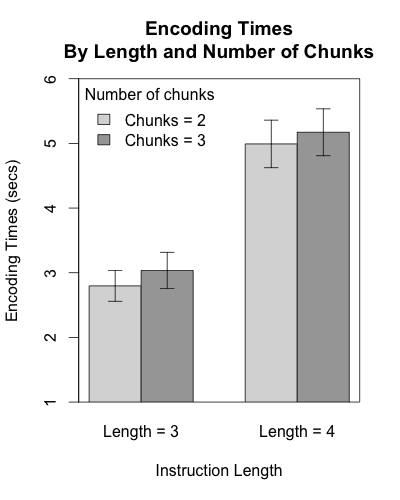


Figure 4. Encoding time by length of instructions (3 vs. 4 operators) and number of chunks (2 vs 3 chunks).

The same analyses were then carried out on the execution times, after similarly removing all the latencies in the top 2.5% of the distribution. The overall pattern of results is shown in Figure 5, which shows the execution times for each of the nine different structures, divided according to the three lengths and the position of the binary operation within the instructions.

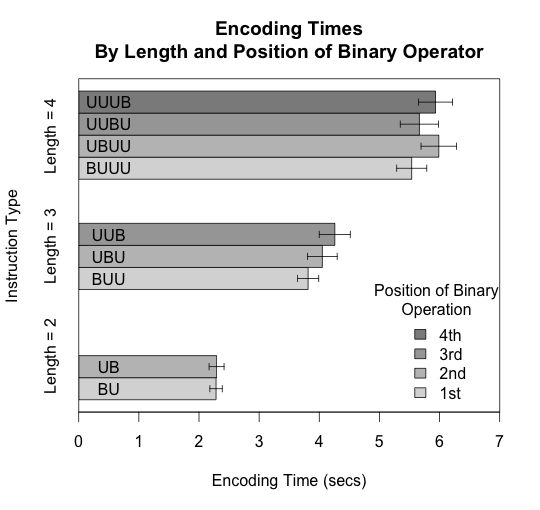


Figure 5. Mean execution times (+/- SEM) for the seven types of instructions examined in this study. Instructions are divided by length (2, 3, or 4 operators) and by the position of the binary operator within the sequence (1st, 2nd, 3rd, or 4th). The corresponding instruction structure (e.g., “UUUB” for instructions of length 4, with binary operation in 4th position; See Figure 1) is indicated on the corresponding bar.

As in the encoding phase, the number of operations had a significant effect on the response times (*F*(2,52) = 500.6, *p* < 0.00001). After excluding those trials whose instructions had only two operators and aggregating the remaining data by length and number of chunks, a 2x2 ANOVA identified only a significant effect of the length (F(1,26) = 453.0, p < 0.00001). Neither the main effect of the number of chunks (F(1, 26) = 0.36, *p* = 0.55) nor the interaction (F(1, 26) = 0.19, *p* = 0.66) were significant. Figure 5 provides a summary of these findings.

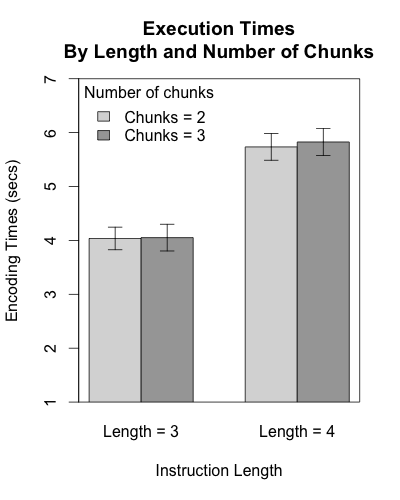


Figure 6. Execution times by length of instructions (3 vs. 4 operators) and number of chunks (2 vs 3 chunks).

**Conclusion and Discussion**

This study investigated the nature of the mental representations that participants create when reading instructions. More specifically, we investigated whether instructions are represented in a “list” format (that is, simply rehearsing the corresponding words) or in a deeper, “structured” format (that is, encoding the specific connections between stimuli and operations, and between different operations). Basing on the findings from the past RITL studies, what kind of forms that these representations take when people learn new tasks has been in a black-box. If representations take the “list” format, length of the instruction becomes the key factor which influences participants’ ability to learn novel instruction. However, if representation takes a structured format, the speed would depend on how the information in the instruction is organized, designed, written, and ordered. To examine this question, we used a recently developed Rapid Instructed Task Learning (RITL) paradigm, which permits to explicitly separate the encoding phase from the execution phase of a single set of instructions. Our results indicate that participants spontaneously divide the instructions into meaningful chunks according to their deep structure, as shown by the increase encoding times when the deep structure was organized in three vs. two chunks, independent of the total number of operations in the instructions. Conversely, execution times were not affected by how instructions were chunked, which is precisely what one would expect if participants had already memorized the way to connect operations in series and properly apply them to stimuli and intermediate results. Finally, we found the significant and consistent result in both length 3 and length4.

These findings may help us reconsider the way to control our performance of learning new instructions. We showed that the speed of learning new instruction is influenced by the deep structure of the instruction and more complex the deep structure become, more time it requires to be encoded. This suggests that there would be a much room of improvement for current instructions used in daily situations. For example, instead of the evaluating the difficulty of an instruction based solely on the length, educators may evaluate the deep structure of the instructions and reconstruct it to make it more feasible while keeping the mass of information same.

A number of limitations need to be acknowledged in our study. First, for experimental constraints, our study employed instructions that were presented in a tightly controlled and artificial format. Specifically, all instructions were given as a series of words naming arithmetic operations and presented as a vertical stack on the screen. Such a format does not necessarily reflect the way instructions are presented in naturalistic settings, such as, for example, school lessons. On the other hand, our presentation is consistent with the procedures used in previous in RITL studies (Stocco, Stocco, Cole, Brass, etc, refs here). More importantly, the fact that instructions were presented as a list was implicitly favoring the alternative, list-format hypothesis, thus making our experimental findings all the more remarkable.

Another limitation of our study is that all the operations used were arithmetic. Because they can be easily combined in different orders, arithmetic operations were instrumental in designing our experimental stimuli. However, arithmetic operations give rise to only a limited and extremely homogenous set of possible instructions. For this reason, further studies will be needed to verify whether our results hold on instructions that comprise different mental operations (such as retrieving information from long term memory, visually scanning an image, mentally rotating a figure) or a combination of mental and motor operations (such as saying out loud words, or planning different hand movements).

A third limitation of our study is that we did not examine the effects of practice on instruction encoding. Previous studies have repeatedly found that practice with the same instructions results in profound effects in terms of both observable behavior and brain activity. In particular, Stocco et al (2012), Stocco and Prat (2014) showed that even limited amounts of practice with the same instructions result in substantially faster encoding times, while the experimental results of Cole et al (2010) suggest that practice significantly altered the patterns of brain activity associated with the encoding of instructions.

These limitations notwithstanding, our results do provide a new and insightful window into the mental processes by which humans encode and memorize instructions. Our results also further prove the utility of studying instructions with RITL paradigms. In fact, to the best of our knowledge, we believe that this study is the very first behavioral-only RITL paradigm, thus proving its usefulness beyond the use of neuroimaging methods.