Listening. Learning. Leading.®

Research Report

ETS RR–12-25

From Biology to Education: Scoring and Clustering Multilingual Text Sequences and Other Sequential Tasks

Jana Z. Sukkarieh

Matthias von Davier

Kentaro Yamamoto

December 2012

ETS Research Report Series

From Biology to Education: Scoring and Clustering Multilingual Text Sequences and Other Sequential Tasks

Jana Z. Sukkarieh, Matthias von Davier, and Kentaro Yamamoto
ETS, Princeton, New Jersey

December 2012

Associate Editors: Joel Tetreault and Keelan Evanini

Reviewers: Paul Deane and Xinhao Wang

Abstract

This document describes a solution to a problem in the automatic content scoring of the multilingual character-by-character highlighting item type. This solution is language independent and represents a significant enhancement. This solution not only facilitates automatic scoring but plays an important role in clustering students' responses; consequently, it has a nontrivial impact on the refinement of the items and/or their scoring guidelines. Furthermore, though designed for a specific problem, the proposed solution is general enough for any educational task that can be transformed into a sequential one. To name a few: It can be used for a set of actions expected from a student in simulations or learning trajectories as projected by a teacher, inside an intelligent tutoring system, or even in a game—or it can simply be used for a set of student clicks, button selections, or keyboard hits expected to reach a correct answer. This solution provides flexibility for existing automatic-scoring techniques and potentially could provide more flexibility if coupled with statistical data-mining techniques.

Key words: multilingual automated scoring, large-scale assessment, sequential tasks, prefix-infix omission and insertion, multilingual character or grapheme sequences, sequence alignment and clustering, bio-NLP

Acknowledgments

Thanks go to Mary Louise Lennon for the description of the item type, pointing out the problems in this task, and answering our questions. We also thank Mike Wagner for explaining the semantics of the attributes in the log files and answering our questions and both Mike and Ramin Hemat for extracting the required data in the log files for all items and languages for this report. Last but not least we thank Harumi Baxter and Monika Kluempen for being such excellent and patient raters.

Table of Contents

List of Tables

List of Figures

There is renewed and growing interest from researchers and assessment organizations in developing types of items that are not multiple choice, whether for monolingual or multilingual assessment. However, it involves considerable time and money to score them manually and the existing methods for scoring automatically have limitations and need improvement. Therefore, we sought to look at a particular item type where we felt we could address these limitations and improve scoring accuracy, with the goal of a solution having broader applications.

This paper is focused on a content-based, character-based highlighting item type with the following specification (Table 1 shows an example in English and Spanish).

For each item, the test taker is given a stimulus and directions to highlight evidence in the stimulus to respond to a certain prompt. A response has a score of true (1) or false (0) or a set of score points that does not necessarily consist of a binary set. The content to be scored is represented in terms of a set of correct responses ranging from the minimum evidence expected to the maximum evidence. In the example, the minimum evidence is given in terms of four pieces: *less than average, fell 4% between 2007-2009*, *fell 7% in 2004*, and *low due to unfavorable economic conditions*. The maximum is a whole section that contains these four pieces. Seen as a set of characters, all correct responses are subsets of the maximum evidence. The complement of the maximum, given the character space to be the stimulus, is what is considered *unacceptable*.

The scoring rules are propositional Boolean logical formulae defined using the *minimum* (these are called *match predicates*) and the *unacceptable* (*mismatch predicates*). A scoring rule is a combination of match predicates and mismatch predicates, where a match predicate, P, is true if and only if P is completely highlighted and a mismatch predicate, Q, is true if and only if any part of Q (any character) is highlighted. In other words, each scoring rule is a function from the set of combination of correct responses (the empty set is false obviously for this particular item type) to the set of score points using propositional logic such as the scoring rule in Table 1.

The rule in the example says that a student's response is true if the pieces of evidence (1), (2), and (4) are completely highlighted and nothing outside the maximum is highlighted or if the pieces of evidence (1), (3), and (4) are completely highlighted and nothing outside the maximum is highlighted. For a true response, a student can highlight up to a whole section that includes the four pieces of evidence defined in the minimum but nothing outside that section.

Table 1

Example of a Highlighting Item

Item 1 (English version)	Item 1 (Spanish version)
Stimulus: (a web page is given)	Stimulus: (un página web es suministrada)
Directions: Look at the web page. Highlight information on the page to answer the following question.	Indicaciones: Lee la página web. Resalte la información en la página que responda la siguiente pregunta.
Question: What does the web page say about the birth rate in USA in comparison to other countries?	Pregunta: ¿Qué dice la página web sobre la tasa de natalidad en USA en comparación con otros países?
Minimum response: 1. Less than average 2. Fell 4% between 2007-2009 3. Fell 7% since 2007 4. Low due to unfavorable economic conditions	Respuesta mínima: 1. Más que el promedio 2. Cayó 4% entre 2007-2009 3. Cayó 7% desde 2007 4. Bajo debido a condiciones económicas desfavorables
Maximum response: Entire section starting with “How birth rate” and ending with “compared to others”	Respuesta máxima: La sección comienza con “como el índice de natalidad” y termina con “comparada con otros”
Scoring rule: 1 if and only if (match(1) AND match(2) AND match(4)) OR (match(1) and match(3) AND match(4)) AND NOT mismatch(max^c); otherwise 0 where max^c is the complement of max (in terms of set theory)	Regla de calificación: 1 if and only if (match(1) AND match(2) AND match(4)) OR (match(1) and match(3) AND match(4)) AND NOT mismatch(max^c); otherwise 0 donde max^c es el complemento de max (en términos de la teoría de conjuntos

In large-scale computer-based assessment, this item type is introduced and highlighting might be enabled one character at a time to allow for multilingual highlighting. In some cases, this item type is administered in 99 natural languages. This character-based highlighting implies that a test taker’s response can unintentionally either have additional characters at the beginning or end of a response or missing characters at the beginning, middle, or end.

For example, in the item illustrated in Table 1, a test taker might miss some characters and end up highlighting something like *ss than average* and *fell 4% betwe 007-200*. Thus, le is missing from (1) and en 2 …. 9 is missing from (2). A student might also highlight the section, "How birth rate … compared to others. How," in which H, o, w were highlighted as additional characters beyond the maximum. A student can add characters at the beginning or end but not in the middle in this particular item type. Table 2 illustrates the various possibilities associated with this item type. We also name each possibility; the names are listed in the first column of Table 2.

Table 2

Omissions and Insertions

Name	Example
Prefix-insertion	***By*** less than average
Prefix-omission	***ss*** than average
Infix-insertion	N/A for this item type
Infix-omission	Fell 4% ***betw*** 2007-2009
Suffix-insertion	Less than average ***in comp***
Suffix-omission	Less than ***ave***

For each highlighted response, any combination of these possibilities could occur, similar to what could occur with a student's written free-text response (except there is no transposition, i.e., no cases where two characters are interchanged—*cieling* versus *ceiling*—and no insertions in the middle).

Being symbolic logic rules, as designed, scoring rules will score responses with additional or missing characters as 0 or false. However, test developers and human raters want to consider these responses true or with a score point 1. The question, obviously, is what would be the admissible maximum number of characters missing or inserted in each case. The answer to this question might vary from one item to another and one language to another. However, as shown later, it is not just about the number of characters.

To summarize the first task that we face:

For each item, given a set of correct responses and a set of unseen responses for a particular multilingual content-based, character-based highlighting item, the aim is to automatically score the unseen responses. We will refer to this task as *prefix-infix-suffix* *omissions and insertions* (for this particular item type, as mentioned, there is no infix insertion, but the aim is a general solution that could work for this and other tasks).

Additional scoring concerns exist. An initial look at the responses revealed that a nontrivial number of students within the same language or across languages responded in a very similar false manner. We noticed that this issue, in general, does not seem to be due to character-by-character highlighting unintentional errors, as seen with the previous issue. Hence, in some cases it made us question the suitability of the wordings in the prompt or the stimulus (such as the passage), its interpretation by students, and the scoring rules, that is, why one response is considered true—or according to the scoring rules can be given a score point of 1—while another that seems to be logically suitable is considered false and can be given a score of 0 only. The issues are particularly complex and challenging when working with different languages, countries, cultures, and background knowledge.

Hence, the second task or question that we need to consider is: Given the set of responses, can we inform the item developers and enhance the item design, including scoring rules developed through what we can automatically mine in the responses? An implicit subtask then is mining the responses automatically and finding similar responses in one language—and across languages if possible.

In the following, we first describe a language-independent methodology to tackle the above two tasks. We then describe some related work. Next we outline the implementation and present the results of the implementation to items and responses written in four natural languages, showing (a) this is a nontrivial enhancement to existing automatic scoring and (b) this solution helps us cluster responses and further enhances the rubrics or the scoring rules—hence, the scoring. Then, we discuss our analysis for the items and corresponding responses and its implications. We conclude with next steps.

Method: A Language-Independent General Solution

Longest Common Subsequence

The basic solution we propose for the above tasks is based on calculating an approximate match or similarity measure between two textual sequences. For the first task, a prefix-infix-suffix omissions and insertions task, a similarity measure between a non-null unseen response and one that is true is calculated. For the second task, any two non-null responses, whether given in the scoring rules or produced by students, are compared via the same similarity measure.

The idea is simple: The closer the similarity—or the further the *Dissimilarity*—between a test taker's response and a correct response, the more likely the response is true. In fact, the closer the similarity (or further the *Dissimilarity*) between any two non-null responses, R_1 and R_2 where R_1 is more likely to be true, the more likely R_2 is true.

In this study, we define the similarity measure between Text_1 and Text_2 to be the longest common subsequences (LCSs; Cormen, Leiserson, Rivest, & Stein, 2001) between them. However, the methodology applied will still hold using other similarity measures.

This particular approach, the use of an LCS to compare test takers' responses, is motivated by biological applications, particularly those used in DNA or gene sequencing, bioinformatics, or genetics.

In biology, a strand of DNA consists of molecules called *bases*: adenine (A), guanine (G), cytosine (C), and thymine (T). A strand of DNA is represented as a *string* over the finite set {A, G, C, T}, where a string is a finite sequence of symbols that are chosen from a set or alphabet. Two strands of DNA, D1 and D2, are similar if they have common bases that appear in the same order but not necessarily consecutively. Hence, this can be seen as generating a third strand, D3, consisting of these common bases. The longer D3 is, the more similar D1 and D2 are. Comparing two strands of DNA in order to verify how close two organisms are is very similar to our task. Figure 1 shows two strands of DNA for two different organisms. When compared to each other, a resulting set of common molecules are shown in the third strand in the figure. Figure 2 shows the same two strands with another longer set of common molecules. By finding similarities between sequences, scientists can infer the function of newly sequenced genes, predict new members of gene families, and explore evolutionary relationships.

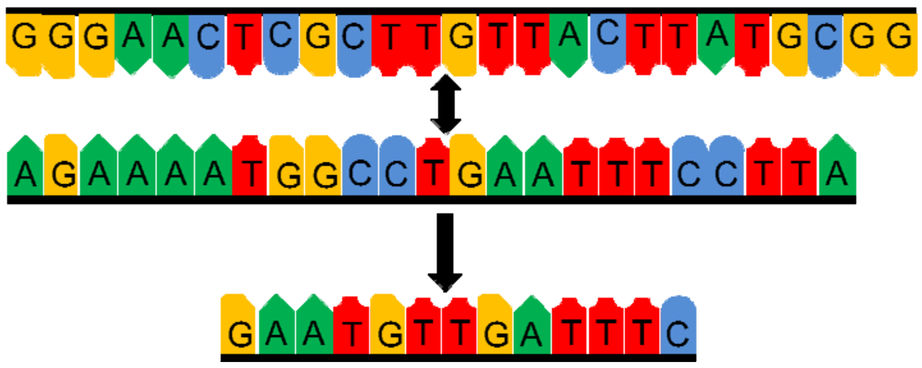


Figure 1. **DNA sequences and commonalities.**

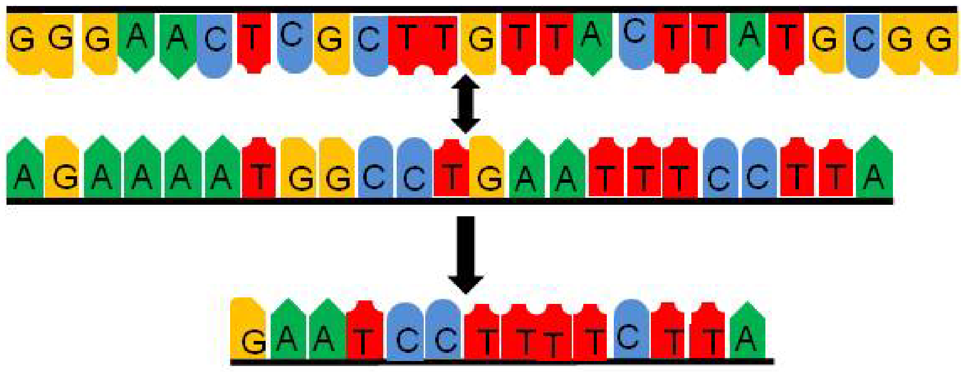


Figure 2. **A larger set of ordered commonalities.**

In general, beyond the DNA and gene alphabet (only 4–20 letters), this idea can be applied to any language. An alphabet of a language can be any set from which the strings of the language may be formed. This set can be finite or infinite, though in most cases it is finite. In our case, it makes sense to say that an alphabet, whether in a formal or natural language, is the set of symbols, letters, or tokens from which the strings of the language may be formed. In other words, a string is a finite sequence or ordered list of elements of an alphabet. A *subsequence* of a string or sequence S is an *ordered subset* (the same order used for the sequence) of the sequence.

A test taker's response can be seen as a strand of DNA or a string over an alphabet. The string, in this case, is a finite set of graphemes where a *grapheme* is the smallest semantically distinguishing unit in a written natural language.[1] A grapheme does not carry meaning by itself. Graphemes include alphabetic letters, Chinese characters, numerical digits, punctuation marks, and the individual symbols of any of the world's writing systems. In the remainder of this document, we will use the term *grapheme* instead of *character* because we are dealing with a multilingual task where student responses can contain numerical digits or symbols. In our case, a subsequence of a string S in any natural language is a set of graphemes that appear in order—the same order as in the writing or script of the language (e.g., in Spanish from left to right, and in Arabic or Hebrew from right to left), but not necessarily consecutively. In a string such as naya, any of na, ay, aa, ya, ny, nay, or nya is an example of subsequences, but not a string such as an or aan.

Given two strings or sequences, a *common subsequence* is one that appears in both, as shown in Figures 1 and 2. An LCS is a common subsequence that has a maximum size or length in terms of number of graphemes. For example, two strings, S1 = jjjsslpljlppjppslppspjljj and S2

= sjsssspjjllpjsspppllpps, have an LCS of jsspllpppplpps. In general, the similarity between two ordered sequences, as defined above, is not necessarily unique. However, its size is. Consider for example, the two sequences, NAY and NYA; there are two LCSs, namely, ny and na. The size or length is 2. Table 3 shows examples of pairs of texts with some LCSs (this particular implementation for results in Table 3 includes white spaces as members of the alphabet).

Table 3

Examples of Pairs of Texts With LCS

Text 1	Text 2	LCS	Length
the Organisation for phony academic publications	the Organization for phony and pub	the Organiation for phony ad pub	32
l'Organisation de faux publications academiques	l'Organzation de pu|lication aca	l'Organation de pulication aca	31
Feliz Año Nuevo! Espero que todos estén muy bién.	Espero verles a todos. Hasta muy pronto.	Espero e todos st muy n.	24
ヨーロッパにおいて、職場でのストレスはわれわれが直面している安全衛生上最大の難問です。調査によれば、欠勤日全体の50%から60%がこれに関連していることを示唆しています。つまり、人の苦痛と経済活動の損失という両方の面から、膨大なコストがかかっているということです。',' す	欧州労働安全衛生機構	安全衛生	4

To find an LCS, we can generate all subsequences and select one with maximum length. Actually, in some cases, the length might be all we are interested in, but as this is anticipated to be a general solution for many item types and additional educational tasks, we want to find the subsequences, too.

Figures 3 and 4 show the two algorithms[2] for finding LCSs and finding the length of an LCS without having to calculate the subsequence, respectively, as described in Cormen et al. (2001).

Let $X = < x_1, x_2, \ldots, x_m>$ and $Y = <y_1, y_2, \ldots, y_n>$ be sequences and let $Z = < z_1, z_2, .., z_k >$ be any LCS of X and Y.

1. If $x_m = y_n$, then $z_k = x_m = y_n$ and Z_{k-1} is an LCS of X_{m-1} and Y_{n-1}
2. If $x_m \neq y_n$, then
   a. If $z_k \neq x_m$ then Z is an LCS of X_{m-1} and Y
   b. If $z_k \neq y_n$ then Z is an LCS of X and Y_{n-1}

***Figure 3.* Calculating an LCS recursively.**

Assume that length (i, j) is the length of an LCS of sequences X_i and Y_j.

$$\text{length}(i,j) = \begin{cases} 0 \text{ if } i = 0 \text{ or } j = 0 \\ \text{length}(i-1, j-1) + 1 \text{ if } i,j > 0 \text{ and } x_i = y_j \\ \max\left(\text{length}(i, j-1), \text{length}(i-1, j)\right) \text{ if } i,j > 0 \text{ and } x_i \neq y_j \end{cases}$$

***Figure 4.* Calculating the length of an LCS recursively without producing the subsequences.**

In summary, we define a similarity between two texts or two test takers' responses in this case, *T1* and *T2*, each seen as a sequence of graphemes, to be

$$Similarity(T_1\,, T_2) = \text{LCS of graphemes between } \mathrm{T}_1 \text{ and } \mathrm{T}_2,$$

where the white spaces and punctuation marks can be included or excluded depending on needs. This function is more meaningful if we were to know the total number of graphemes in both T_1 and T_2. Let [*H*] denote the number of graphemes in a sequence of graphemes *H*. We calculate more indicative measures:

$$NSimilarity(T_1\,, T_2) = \frac{[Similarity(T_1\,, T_2)]}{[T_1\,] + [T_2]}$$

or

$$Dissimilarity(T_1\,, T_2) = ([\,T_1] - [Similarity(T_1\,, T_2)]) + ([\,T_2] - [Similarity(T_1\,, T_2)]).$$

Obviously, the smaller the *Dissimilarity* measure, the more the similarity between the two sequences of graphemes. Again, normalizing the above measure using the total number of graphemes in the two sequences will be:

$$NDissimilarity(T_1, T_2) = \frac{([T_1] - [Similarity(T_1, T_2)]) + ([T_2] - [Similarity(T_1, T_2)])}{[T_1] + [T_2]}$$

$$= \frac{[T_1] + ([T_2] - 2\,[Similarity(T_1, T_2)])}{[T_1] + [T_2]}$$

$$= 1 - \frac{2\,[Similarity(T_1, T_2)]}{[T_1] + [T_2]}$$

$$= 1 - 2\,NSimilarity(T_1, T_2).$$

We recommend normalizing the measures because the application is going to be over a varied number of responses, items, and natural languages. Note that we can define *Dissimilarity*($T1,T2$) as $[T1]$ + $[T2]$ - *Similarity*($T1,T2$), which makes *NDissimilarity*(T_1,T_2) simply *1-NSimilarity*(T_1,T_2). However, taking different combined measures into consideration is helpful and empirical evidence will confirm their adequacy. Having defined an approximate match or similarity measure, its use to solve our concerns is described in the following section.

Scoring Sequences

The first task is to automatically score unseen responses given a set of correct responses. Consider the set of non-null test takers' responses where each response is seen as a sequence of graphemes (denote this set by $ResUnseen$) and consider the set of distinct responses that are also correct or true where each response is seen as a sequence of graphemes (denote this set by $ResTrue$), $\forall\, xu_i \neq \emptyset$ such that $xu_i \in ResUnseen$ & $\forall\, yt_j$ such that $yt_j \in ResTrue$, calculate a four-tuple:

$$A_{ij} =< \text{LCS}(xu_i, yt_j), \text{Length\_of\_LCS}(xu_i, yt_j), Dissimilarity(xu_i, yt_j), NDissimilarity(xu_i, yt_j) >.$$

For each item, an mxn matrix, Z, is obtained where m is the number of non-null responses and n is the number of distinct true responses:

$$\begin{pmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & \ddots & \vdots \\ A_{m1} & \cdots & A_{mn} \end{pmatrix}.$$

One way of reducing the dimensions of Z is to collapse similar responses into uniform or canonical representations, such as removing white spaces and punctuation marks. This means that only the uniform representations of xu_i and yt_j need to be compared.

Figure 5 shows the comparison of each non-null response and each true response. Zt* denotes row t in Z matrix. For the ith non-null response, first find the minimum *Dissimilarity*, Δi $= \min_j\{NDissimilarity(yt_j, xu_i)\}$.

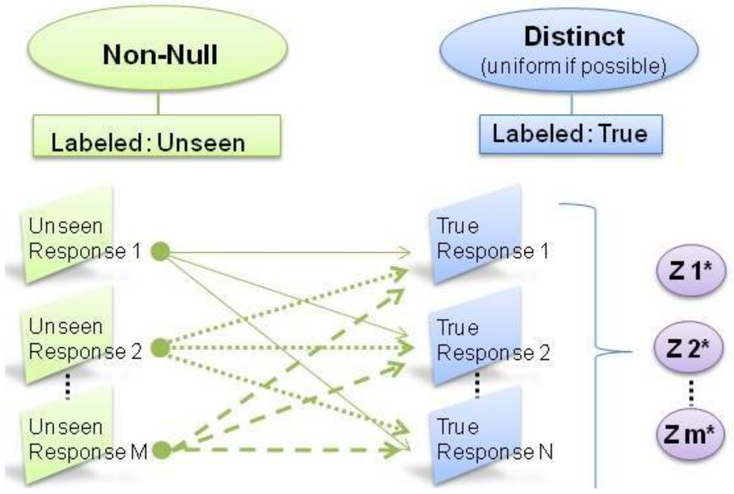


***Figure 5.* Compare each unseen to each true.**

The same logic applies if we were to choose *Similarity*, but in that case we will choose the maximum instead of minimum. Thus, Δ_i is the minimum d_{ij} over all four-tuples in row Z_{i*}. For simplicity, we will denote the arguments of A_{ij} with $<a_{ij}, b_{ij}, c_{ij}, d_{ij}>$. Hence, Δ_i is of the form d_{ik} for some k between 1 and n, that is, for some true response. Hence, Z's dimensions get reduced to m-by-1, and an m-by-1 column vector is obtained, as seen in Figure 6. Note that the selection of the minimum or maximum is the most intuitive and that is what we use in this study.

However, it is not the only option; one might, for example, select a linear function of the various (dis)similarities with weights corresponding to the number of repeated true responses.

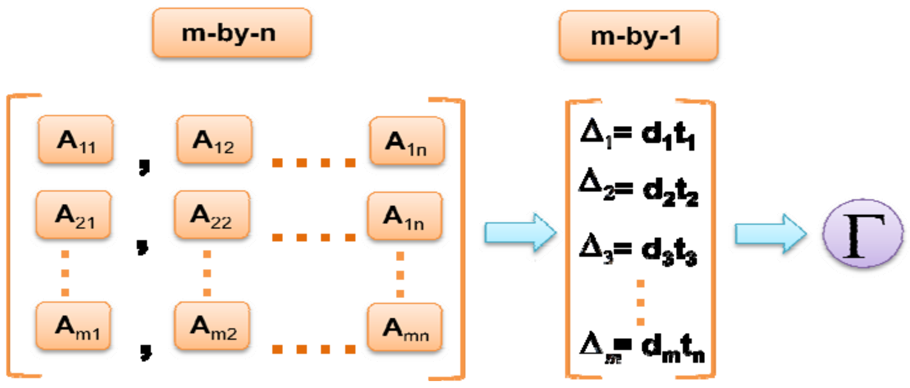


***Figure 6. m*-by-*n* matrix to *m*-by-1 matrix.**

However, at least for each item, the aim is to have one threshold or a tolerance, Γ, for *Dissimilarity* with what is considered correct or true, such that if $\exists\ dij \leq \Gamma$, then the score of xu_i is true. Hence, the *m*-by-1 column vector will guide our choice of the threshold, Γ. In the following, we describe how.

Selecting a Threshold

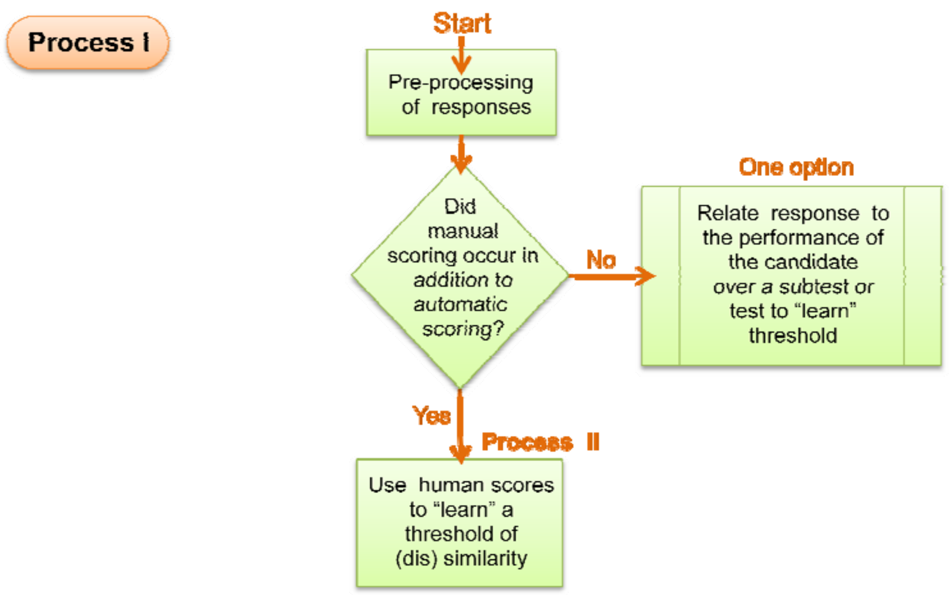
One option is just to set a threshold based on statistics related to the number of graphemes over all responses or the minimum and maximum correct responses (when these are available) where its selection is general across all items. A crude option, for instance, independent of any data, is to select a random threshold such as 0.5 or use some heuristics over *NDissimilarity*. For instance, select various thresholds that satisfy an arithmetic progression of the form $\Gamma_n = \Gamma_m + (n - m)\ d$ over all items across all languages or a subset of items over a subset of languages. For example, starting with a threshold of 0.1 and difference *d* of 0.1, the set of thresholds to consider would be {0.1, 0.2, ... , upper bound of *NDissimilarity*}.

Another option is to use a set of training or field-test data to guide our selection of a tolerance of *Dissimilarity* that can be applied on an unseen set of responses. Whether human scoring takes place for the training dataset or not will have an impact on threshold selection. Figure 7 shows that if human scores are given for each item, then we use the human scores to learn a threshold of (dis)similarity. If not, then one option might be to define a one-to-one correspondence between a response produced by a candidate, C, and C's performance over a subtest of items belonging, for example, to the same stimulus or the whole test, in order to learn a threshold. In any of these cases, a machine-learning algorithm also can be used to learn a threshold-given item-related or response-related attributes, such as LCSs, *Dissimilarity*, and so on. In all the mentioned approaches for a threshold selection, using a training dataset, that is, a set of student responses labeled or scored manually, is considered a supervised approach, while the rest are unsupervised approaches for learning or selecting a threshold.

In this report, we concentrate on this first case, that is, where human scores are given with the responses. The other case where human scores are not given will be described in a separate study.

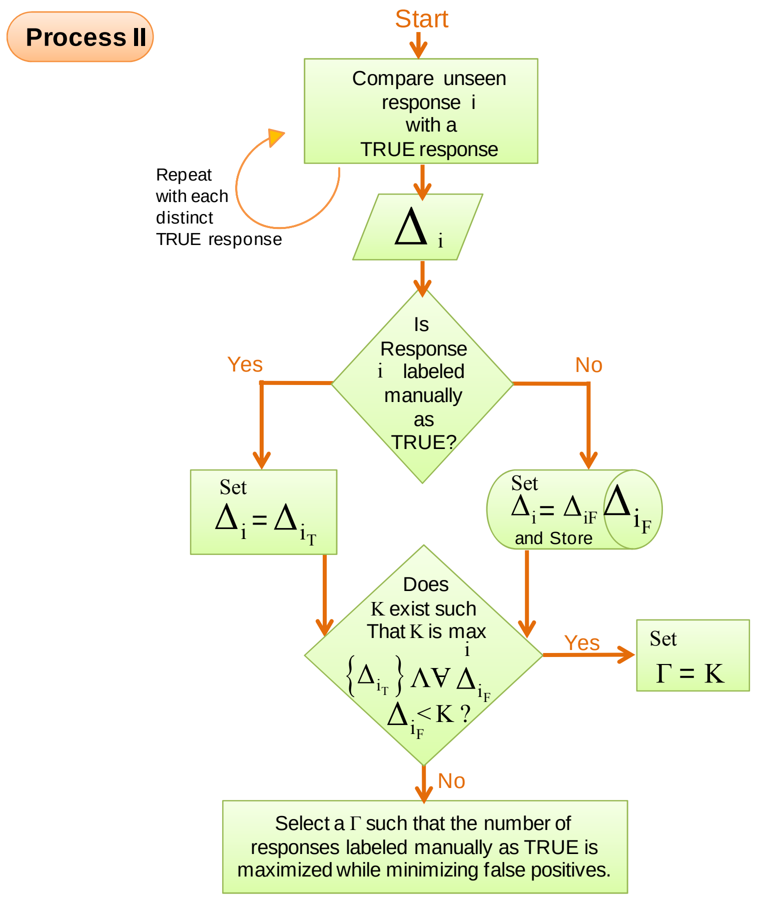
In Figure 7, a preprocessing step is introduced. This step might simply be extracting relevant information from the log files or database used to store student responses or it could be removing white spaces and punctuation marks from student responses to obtain a uniform representation for a response or a sequence, and so on. In the evaluation section of this paper, the preprocessing step for this study will be specified.

Figure 8 summarizes the process by which a threshold is selected given a set of student responses with their corresponding human scores (used as a training dataset). For each item, once the above matrix *Z* is obtained and reduced to the *mx1* vector of Δ_i *s*, as described above, then $\forall\ xu_i \neq \emptyset$ in the training dataset, the question is whether xu_i is scored manually as true or not. If it was true, then denote $\Delta_i = \Delta_{iT}$, and if not, then denote $\Delta_i = \Delta_{iF}$. Find *K* such that *K* is $\max_i\{\Delta_{iT}\} \wedge \forall\ \Delta_{iF}.\ \Delta_{iF} < K$. If such a *K* exists, then set the threshold $\Gamma = K$. In practice, in this case, Γ is really one of the Δ_{iT} s. If such a *K* does not exist, then select a Γ such that the number of responses labeled manually as true will be maximized while minimizing the number of false positives, that is, responses labeled manually as false that we do not want to wrongly score as true. Once a threshold is selected over a certain item and non-null response dataset, then this threshold can be used to score future unseen responses.



Process I
Start
Pre-processing of responses
Did manual scoring occur in addition to automatic scoring?
No
One option
Relate response to the performance of the candidate over a subtest or test to "learn" threshold
Yes
Process II
Use human scores to "learn" a threshold of (dis) similarity

Figure 7. **Did human scoring occur?**



Process II
Start
Compare unseen response i with a TRUE response
Repeat with each distinct TRUE response
Δ_i
Is Response i labeled manually as TRUE?
Yes
No
Set $\Delta_i = \Delta_{i_T}$
Set $\Delta_i = \Delta_{iF}$ and Store Δ_{i_F}
Does K exist such That K is $\max_i \{\Delta_{i_T}\} \wedge \forall \Delta_{i_F}$ $\Delta_{i_F} < K$?
Yes
Set $\Gamma = K$
No
Select a Γ such that the number of responses labeled manually as TRUE is maximized while minimizing false positives.

Figure 8. **Selecting a tolerance for *Dissimilarity*.**

Clustering Sequences

Calculating similarity measures allows us to cluster sequences or students' responses. For each item, the similarity measure is calculated between all responses (given in the scoring rules or written by students). The matrix that we calculated earlier in Figure 5 will become square or an *m*-by-*m* matrix where m is the number of all available (possible) responses. In the case of LCS and the fact that it is a commutative relation, the *m*-by-*m* matrix obtained is symmetric with the diagonal elements equal to each other.

Next, the responses are clustered into equivalence classes depending on their proximity to each other. The clusters of responses might be true with high certainty (green or a walking-person symbol) because of their proximity with correct or true responses, false with high certainty (red or open-arm person symbol) because of its remoteness to a true response, and uncertain (yellow or person-with-arms-down symbol) and need human intervention. This process can be made adaptive or dynamic in the sense that once the representative of the equivalence class or the cluster is labeled/scored (whether manually or automatically), a new unseen response can either fit into an already existing cluster or create a new cluster of its own (yellow) and the process can be repeated. Figure 9 illustrates the idea.

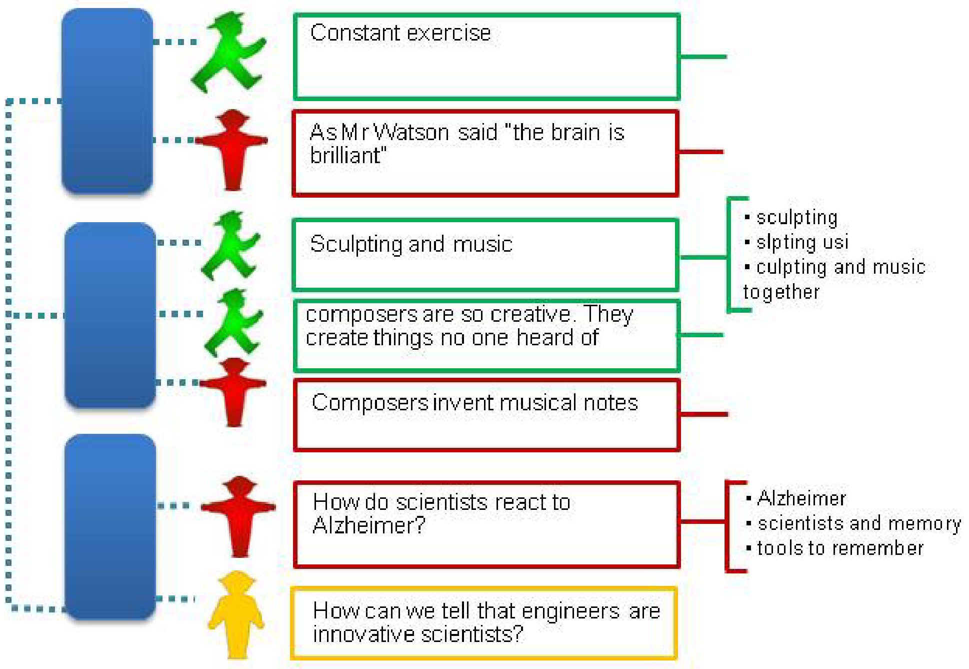

Constant exercise
As Mr Watson said "the brain is brilliant"
Sculpting and music
composers are so creative. They create things no one heard of
Composers invent musical notes
How do scientists react to Alzheimer?
How can we tell that engineers are innovative scientists?
• sculpting
• slpting usi
• culpting and music together
• Alzheimer
• scientists and memory
• tools to remember

Figure 9. **Response clustering.**

In the illustration in the figure, there are three clusters of responses, each corresponding to an item or a particular language for example, and each cluster has subclusters of green, red, and yellow labels. The first subcluster consists of true responses and the second consists of false responses. The text that appears in the box is one representative response of the cluster or equivalence class. If we were to click on the box, we would get a list of all responses corresponding to that cluster. As mentioned above, an unseen response can either fit under any of the existing clusters if it is similar to the members of that cluster/equivalence class or create a cluster of its own, which by default is labeled as *not_labeled* (yellow). In that case, a human can look at that newly formed cluster and change its label to red or green.

Such clustering of responses corresponding to one item in one language or across languages has a nontrivial impact on content design, representation, and the scoring rules associated with this content. In the evaluation section, we will describe its impact and provide concrete examples.

Before we go on to present the related work, implementation, and evaluation sections, we need to describe two methods that seem to be, at first glance, good solutions for this specific problem of the highlighting items. These methods are *difference or commonality in terms of number of graphemes* and *LCS*. We argue, however, that though they work for some cases, they are not general solutions, while the LCS-based solution is.

A naïve baseline that compares the length of a non-null unseen response (UR) to the length of each true response (TR):

$$\text{Normalized difference} = \frac{abs\,([\text{UR}] - [\text{TR}])}{[\text{UR}]+[\text{TR}]}\ .$$

This comparison of length is not a general solution for the problem, though it seems like it works in some cases. Consider the following example, where it seems acceptable to score the unseen response as true:

- Unseen response: Sci discover t world that exists; engineers create the world that nev
- True response: scientists discover the world that exists

$$\text{Normalized difference} = \frac{abs(69 - 42)}{[69] + [42]} = 0.24.$$

However, consider the following example where it is not acceptable to score the unseen response as true:

- Unseen response: fire lookout towers roads water supply systems water drainage systems airports bridges sports stadiums toilet blocks marinas towers for wind generators elevated viewing platforms in forests or above lakes In short, engineers create the sorts of things we use all the time. Civil engineers can show their work to others with pride. And when these structures need repair or modification or when they've outlasted their usefulness, and need to be removed, these operations are the responsibility of civil engineers as well.
- True response: What's the difference between a scientist and an engineer? The well-known physicist Theodore von Newman once said, "Scientists discover the world that exists; engineers create the world that never was." Or to use a literary metaphor, scientists write about the rules of poetry, whereas engineers write the poetry itself.

$$\text{Normalized difference} = \frac{\text{abs}(522\text{-}321)}{[522] + [321]} = 0.24.$$

Both have a normalized difference of 0.24. Hence, it is important to know if there is any commonality and not just a count. This does not mean that there might not be a false positive (i.e., a false response scored wrongly as a true response) using the LCS-based method once a threshold is selected, but it is more likely that there will not be as many relative to the naïve length comparison.

With the second method, LCS, the question is whether we can consider a substring and not a subsequence an approximate match measure, that is,

$$\mathit{Similarity}(yt_j\,, xu_i) = \text{LCS of graphemes between } yt_j \text{ and } xu_i.$$

The difference between a substring and a subsequence is that in a substring, only consecutive characters are considered, not just characters in the same order or direction of writing the language. This will also work in many cases, but it will not account for infix omissions. Consider the true response "quick dirty work," and consider the student response "quck drty wok"; an LCS will be rty wo of Length 5 (excluding white spaces) and an LCS would be quck drty wok of Length 11 (excluding white spaces). On the other hand, accounting for infix omissions might backfire, too, in some cases. Consider the true response "quick dirty work," and consider the student response "life can be dandy, wild and honorable." Then, an LCS would be i c d w or of Size 6 (excluding spaces) and an LCS would be a maximum of Length 2.

A comparative evaluation between LCSs and substrings depends on the type of data in consideration and can only be shown empirically. However, intuitively, finding the LCSs is a more general and correct solution than finding the LCSs for our tasks.

Related Work

The medical literature has a lot to say about sequences and their comparisons. Sequence comparison can be done in two ways: *qualitatively*, or visually, and *quantitatively*. One type of comparison is known as *alignment*. It consists of two components: defining a similarity measure and defining an algorithm that will find the optimal alignment—in other words, defining a scoring scheme, scoring all sequence alignments, and then selecting the alignment with the best score. For example, in our case, the similarity measure is defined as the length of the LCSs; we assume different heuristic approaches to find the optimal alignment or select a threshold.

In simple terms, sequence alignment is the most economical method to transform a sequence into another. For example, a sequence like ABCD is transformed to EBCD by substituting A with E, while to transform it to EBD, an additional deletion of C will be required. Hence, assuming all operations cost the same, the cost of aligning ABCD with EBCD is less than aligning it with EBD. The operators do not have to be restricted to deletion or omission or to substitution and transposition (called *indels*) and they do not necessarily have equal cost. In general, given a set of operators over a set of alphabet, one can define a cost for transforming one into another.

There are different types of sequence alignment: *global*, *local*, and *multiple sequence alignment*. Global alignment is the best alignment over the entire length of the two sequences. It usually starts at the beginning of the two sequences and adds gaps to each until the end of one is reached. Local alignment refers to considering alignment over subsequences and not the entire length of the two sequences in question. It finds the regions of highest similarity between the two sequences and builds the alignment outward from there. Multiple sequence alignment involves more than two sequences. We will not go into it in this particular study. Table 4 lists some of the well-known similarity measures, scoring schemes, and the algorithm(s)—local or global—used to find the optimal alignment.

A brute force approach to find optimal alignment would be to generate all alignments, score them, and select the best score—an approach that is very impractical. The Needleman-Wunsch approach (Needleman & Wunsch, 1970) reduces the number of possibilities considerably yet guarantees that the best solution is still obtained. The basic idea is to build up the best alignment by using optimal alignments of smaller subsequences. Sellers (1974) used a metric distance to define similarity and select the best score. This led to the Smith-Waterman algorithm (Smith & Waterman, 1981; Arratia & Waterman, 1994), the most accurate in database search but the slowest. Since then there have been many variations for optimizations and more efficient algorithms. For example, Basic Local Alignment Search Tool (BLAST; Altschul, Gish, Miller, Myers, & Lipman, 1990; Altschul et al., 1997) is the tool most frequently used for calculating sequence similarity. BLAST was developed to provide a faster technique than another algorithm developed earlier, called FAST-All (FASTA; Pearson & Lipman, 1988). FASTA, which works with any alphabet, is an extension of two other tools, FAST-P (for protein) and FAST-N (for nucleotide).

The *edit distance* (Levenshtein, 1965, 1966) is the number of deletions, insertions, or substitutions required to transform a string S into a string T. Several variations of this metric exist; for example, the *Damerau–Levenshtein distance* (Damerau, 1964). accounts for number of transpositions of two adjacent characters, too. The cost of each operation—insertion, deletion, transposition, substitution—might vary.

One can easily see that any of these sequence alignment techniques can be seen as a variation of looking at our problem. In particular, when only insertion and deletion (no substitution) are allowed or when the cost of substitution is double the cost of an insertion or deletion, then for two sequences $S = s_1, s_2, \ldots s_n$ and $T = t_1,t_2, \ldots t_n$ the edit distance$(S,T)= n + m - 2$ LCS(S,T). Hence, one can easily make a comparative evaluation with the edit distance measure under these conditions. However, the reason the edit distance measure was not selected first for this particular application is that it is not always commutative. It is only commutative when the cost of each operation is the same. We wanted to start with a measure that will partition the space of responses into equivalence classes without additional conditions.

Table 4

Some Well-Known Alignment Techniques

Finding optimal alignment	Type	Comments
Needleman-Wunsch (1970)	Global alignment	Same accuracy as Smith-Waterman. Slower than BLAST but faster than Smith-Waterman
Smith-Waterman (1981) or better implementations of it (Gotoh, 1982; Altschul & Erickson, 1986)	Local sequence alignment	Used in FASTA Slower than BLAST but more accurate (exhaustive and uses dynamic programming). It is a variation of the Needleman-Wunsch.
BLAST: Heuristic approach approximating the Smith-Waterman algorithm (Altschul et al., 1990)	Local	BLAST Faster than Smith-Waterman but less accurate
Align	Global	FASTA Implements the Needleman-Wunsch global alignment algorithm
LAlign	Local: Huang and Miller (1991)	FASTA Implements the Waterman local-alignment algorithm FASTA uses a hybrid of heuristic and exhaustive approaches.
Needle	Global	EMBOSS Same program as Align, but it has an improved version called Stretcher (Myers & Miller, 1989)
Edit-distance measure	Global	Several tools
LCS	Can be used both global and local	Several tools including ours

Sequence alignment is used beyond biology and bioinformatics. In fact, some of these computer science techniques might have been originally developed for text editing and text comparison but became more popular with biological applications. For example, file comparison programs such as the function *diff* in Unix that compares pairs of lines belonging to each file, in a sequential order, uses the LCS algorithm. The diff program was originally written by D. McIlroy and J. W. Hunt. Their implementation was for an algorithm originally published by Hunt and Szymanski (1977).

In linguistics, sequence alignment has been used in various areas. For instance, in natural language generation, sequence alignment techniques were used to produce linguistic versions of computer-generated mathematical proofs (Barzilay & Lee, 2002). Spell-checking techniques could depend on sequence alignment (Sasu, 2011), and similarly for speech recognition (Pucher, Turk, Ajmera, & Fecher, 2007, Ziółko, Gałka, Skurzok, & Jadczyk, 2010). In fact, many sequence algorithms had extensive use and some success in speech recognition since the early 1990s. Another application has been plagiarism detection (Lukashenko, Graudina, & Grundspenkis, 2007; Su et al., 2008). In comparative linguistics, sequence alignment was used to compare or reconstruct languages automatically (Kondrak, 2002).

Furthermore, alignment techniques were used in business applications—for example, to analyze purchases over time (Prinzie & Van den Poel, 2005).

This Study

Description

For this particular study, we didn't have access to the minimum evidence, maximum evidence or scoring rules, only the automatic scores obtained using symbolic logic strict rules. Also, beyond the minimum, missing characters within the maximum were tolerated in the existing implementation of the scoring rules. For instance, a response, "Hw birth ra less than average fell 4% between 2007–2009 low due to unfavorable economic conditions," for the item in Table 1 is scored automatically as true while "Hw birth ra less than averag fell 4% between 2007–2009 low due to unfavorable economic conditions" is scored as false. Hence, the scoring task was reduced to the following: For each item, given the responses and their scores, where scores have been obtained automatically using the propositional logical rules, the aim is to correct the scores of responses scored as false due to insertion or omission of characters.

The processes and methodologies were applied as described earlier in the method section. The set of non-null test takers' responses that were scored automatically as false were considered instead of the set of unseen responses, $ResUnseen$, and the set of distinct responses that were scored automatically as true were considered $ResTrue$.

The clustering task was reduced to the available responses with no access to true responses except the ones scored automatically as true. In both tasks, the assumption was that responses scored automatically as true are scored correctly.

Implementation

This is an intuitively simple problem, but surprisingly difficult to solve and implement correctly. Methods to improve performance efficiency in terms of time and memory can make a difference for implementation, as the literature suggests. In the following, we provide a complexity analysis for each item that can be easily generalized to any number of items.

The preprocessing step includes (a) finding all responses that were labeled automatically as false, (b) finding all responses that were labeled automatically as true (and any true responses if we have access to the minimum and maximum), (c) filtering out all null responses labeled as false, and (d) considering distinct true responses, and then transforming them to a uniform representation. Hence, same or uniformly represented responses are identified and only one is considered in the processing. Also, we applied some quality assurance steps, such as no null responses being scored as true, and each candidate has one and only one response (the latest he or she produced).

Transforming the non-null responses labeled as false to a uniform representation is performed similarly to true responses for consistency and efficiency, keeping in mind that candidate IDs for responses labeled as false have to be possible to retrieve. Also, the results for each pair true response (yt_j) and false response (xf_i): <Text1/xf_i, Text2/yt_j, LCS, Length, *Dissimilarity*, and *NDissimilarity*> can be stored in a database in order to avoid recalculating for the same pair of sequences or texts. If Text1 and Text2 have already been encountered, we just look them up.

Hence, in practice, one would be dealing with *M* distinct uniformly represented non-null responses labeled as false and *N* distinct uniformly represented true responses.

All of the above preprocessing steps are either constant or linear in the number of responses labeled as true or false. Hence, this preprocessing step will be ignored as part of the analysis of runtime. Also, once the *M* x *N* matrix is produced, calculating Δ_i and selecting the threshold also takes a constant time. The runtime of calculating the matrix, in the worst case scenario, is:

$$F(Z) = M \text{ x } N\, O(LCS(T_i, T_j)) \text{ where } I = 1, 2, ..., m \text{ and } j = 1, 2, \ldots, n.$$

Hence, the process of comparing two texts, that is, finding the LCSs, is what is crucial to analyze. This is a classical computer science problem and has been studied extensively. Many suggested implementations with various complexities exist in the literature (Hadlock, 1988; Hirschberg, 1977; Hunt & Szymanski, 1977; Masek & Paterson, 1980; Myers, 1986; Nakatsu, Kambayashi, & Yajima, 1982; Wagner & Fischer, 1974).

As seen in Table 4, the algorithm is recursive. However, if it would be implemented just recursively by calculating all subsequences and selecting one, then its runtime in the worst-case scenario would be exponential in the number of characters in the two texts. If implemented in a dynamic way or using memorization (Wagner & Fisher, 1974), then it has an $O(n\text{x}m)$ worst-case running time where n and m are the number of graphemes in each sequence. Its space requirement in the worst case is quadratic, too. Hirschberg's algorithm (1975) reduced the space requirement to $O(n + m)$. Later, an improvement to $O(\frac{nxm}{\log n + m})$ was suggested by Masek and Paterson (1980).

Other existing algorithms present complexities that depend on parameters other than n. For example, Myers (1986) and Nakatsu et al. (1982) suggested an algorithm with $O((n + m)\, D)$ where D is the simple Levenshtein distance between two given strings. Iliopoulous and Rahman (2008) suggested an algorithm with $O((n + m)\, R)$ where R is the total number of ordered pairs of positions at which the two sequences match. In 2011, Thang (2011) suggested that if the two sequences belonged to two finite languages accepted by two finite automata, $A1$ and $A2$, then the algorithm of finding the LCS is $O(|A_1|\, |A_2|)$ worst-case running time, where $|A_i|$ is the number of states and edges of automata A_i.

In our case, for the first prototype, we implemented a dynamic recursive algorithm in *Sociaal-Wetenschappelijke Informatica* (SWI) Prolog[3] with a quadratic time and quadratic space requirement. Graphemes were transformed to Unicode (utf8) representation to make the implementation natural language-independent.

In an operational setting, we need only one true response with *Dissimilarity* less than or equal to the threshold for the false responses to switch to a true response.

As mentioned earlier, for efficiency, the results for each pair—Text1/xf_i, Text2/yt_j, LCS, Length of LCS, *Dissimilarity*, and *NDissimilarity*—have been stored in a database in order to avoid recalculating for the same pair of sequences or texts. If Text1 and Text2 already have been encountered, we just look them up.

Evaluation

For the evaluation of the approach and techniques, four highlighting items belonging to the item type previously described, written in English (EN), Spanish (ES), French (FR), and Japanese (JA), were considered. The language codes were adopted from Wikipedia ("List of ISO 639-1 codes," n.d.). All responses in all languages were scored manually. All English, French, and Spanish responses were scored by the same human rater. All the Japanese responses were scored by one human rater, who was different from the person who scored the English, French, and Spanish. The responses were collected in Chile (CL), Canada (CA), Britain (GB), France (FR), Spain (ES), Ireland (IE), and Japan (JP). The country codes were adopted from Wikipedia ("ISO 3166-2," n.d.).

Results

We first include some average sequence length for different languages in each item, which illustrates how sequences considered in this task are by far much shorter than sequences considered in DNA sequence algorithms in general. It also gives an idea on the range of true student responses. Figure 10 summarizes the range and the average number of graphemes of non-null false and true responses. In general, on average, the number of graphemes for responses scored automatically as false is larger than that of responses scored as true.

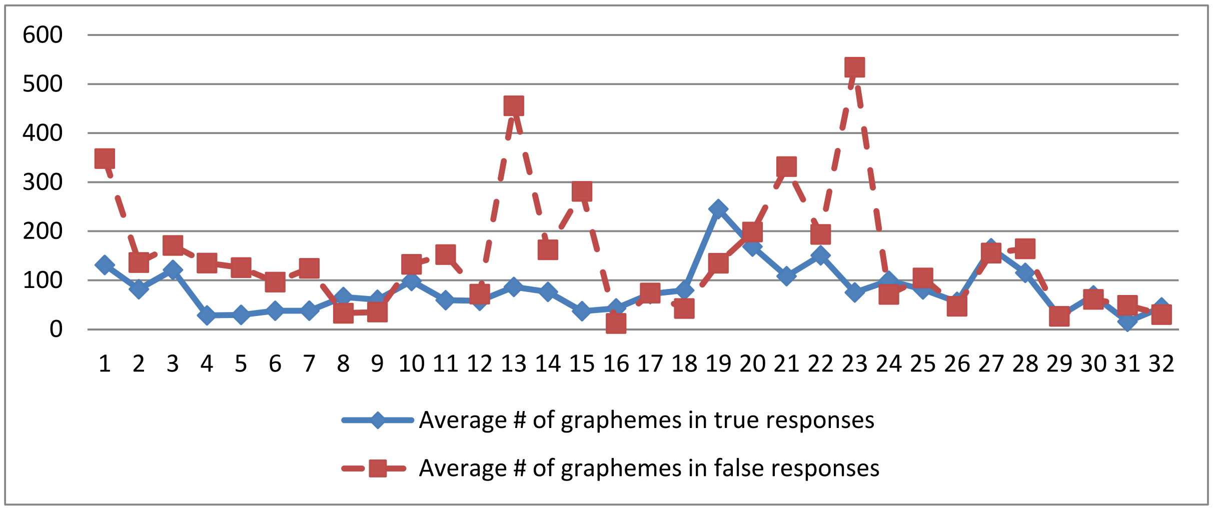


Figure 10. **Average number of graphemes in the 32 <country, language> considered.**

In the following, we will present the results of our scoring approach for each item separately.

Because there is very little data, we decided to train on responses associated with one country only, regardless of the size of the dataset associated with that country and its language. Once a threshold is selected based on that country, the evaluation is performed using the datasets of responses corresponding to the rest of the countries even if the datasets belonged to other languages. This assumes that the rubrics for each item in all countries are defined in the same way. There was no specific methodology for selecting the set of training data except the order of the files listed. In other words, there is no intuitive reason for favoring one country or language over the other. Hence, the methodology could be modified when more data are available. For instance, the experiment could be repeated by dividing the number of responses equally or by treating each language separately because some issues might be language-specific issues.

In the rest of the document we will refer to automatic scores based on the symbolic logical rules given with each item as original automatic scores and the scores based on our LCS method as updated automatic scores. Recall that there are several assumptions. First, there is no access to the scoring rules. Second, the original true responses are correct. In fact, there is absolutely no reason why any of them should be incorrect because the existing Boolean implementation was verified. The aim is to rescore the set of non-null responses with original scores false.

Tables 5–16 consist of three tables of results for each of the four items. The first table for an item contains (a) the total number of student responses, (b) the number of responses with original automatic scores that are false, (c) the number of non-null responses with original scores that are false, (d) the number of non-null distinct responses with original scores that are false, (e) the number of non-null distinct uniform responses with original scores that are false, (f) the number of responses with original automatic scores that are true, (g) the number of distinct responses with original scores that are true, and (h) the number of distinct responses represented in a uniform way with original scores that are true. Recall that a uniform representation is one with no punctuations and no spaces. Some languages, such as Japanese, have fewer punctuation marks than other languages and graphemes are not separated with white spaces; hence, a uniform representation might not make much difference.

The second table for an item contains (a) the number of non-null response with a human score of false with an original automatic score that is false, (b) the number of non-null false responses labeled by humans as true, (c) the number of responses scored by humans as false that are corrected properly to true using our scoring method, and (d) the number of responses scored by humans as false but wrongly scored by our method as true.

The third table summarizes the results for each item in terms of accuracy, precision, recall, and true negative rate. The meaning of precision = 1 and recall = 1 is that the method used has yielded the whole truth and nothing but the truth. In our case, precision = 1 means that the new automatic scoring method is definitely an improvement over the existing technique, that is, the implementation of the Boolean scoring rules.

Results for Item 1

The following three tables of results correspond to Item 1.

Table 5

Item 1: False Versus True Responses as Originally Scored Automatically

	Total	False	Non-null false	False distinct non-null	False distinct non-null uniform	True	True distinct	True distinct uniform
IE_EN	85	52	42	32	30	33	14	5
GB_EN	69	53	38	26	26	16	11	6
CA_EN	48	30	19	17	17	18	9	6
CL_ES	94	70	44	34	32	24	9	4
ES_ES	127	90	58	40	40	37	16	10
FR_FR	244	164	108	85	82	80	29	27
CA_FR	66	41	29	19	19	25	13	11
JP_JA	153	75	50	43	43	78	23	23

Training on IE_EN, a threshold of 0.19 is learned based on the responses labeled by humans. Table 6 shows the results for the evaluation using the datasets corresponding to other countries and languages with the threshold of 0.19. In GB_EN, five of six are scored correctly to true with the threshold selected. One response has a minimum *Dissimilarity* of 0.21 with a true response; hence, it is missed being scored correctly. CA_EN has zero true responses labeled by humans. In ES_ES, the three responses get corrected. Actually, it turns out they only need a threshold of 0.07. In FR_FR, nine of the 12 pass, but three responses need minimum dissimilarities of 0.2, 0.22, and 0.25, respectively, to be corrected. The eight responses in CA_FR pass and get scored correctly. In fact, a threshold of 0.16 turns out to be enough for these responses. For JP_JA, only three of the 16 responses pass the threshold, while each of the rest has a maximum *Dissimilarity* with a true response greater than 0.19.

Table 6

Item 1: How Many False Get Corrected Properly to True?

	Non-null with original automatic score as false	Non-null original score false scored by humans as true	Non-null original score false with human score true updated automatic score as true	Non-null original score false with human score false with updated automatic score as true
IE_EN (training)	42	3	3	0
GB_EN	38	6	5	0
CA_EN	19	0	0	0
CL_ES	44	1	1	0
ES_ES	58	4	4	0
FR_FR	108	12	9	0
CA_FR	29	8	8	0
JP_JA	50	16	3	0

One of the false responses in CL_ES that was labeled by humans as false and was about to pass the threshold (0.199) was the following. The student highlighted the true response but some additional information was also highlighted that—though it kept it in the proximity of true—was a part of a quote from someone who is different from the person in the prompt of the item:

> s. El entrenamiento específico mejora la atención dividida," señala Sekuler…"Una sesión de entrenamiento de tan solo dos horas mejora los resultados de las pruebas de las personas mayores. El entrenamiento sostenido puede hacer maravillas"

This is one case where the context, or more simply the position of the graphemes in the passages, can be used to help. Table 7 summarizes the results.

Table 7

Item 1: How Many False Get Corrected Properly to True? (Summary)

Item	Training of positive instances (# of negative instances)	Evaluation	Accuracy	Precision	Recall	True negative rate
Item 1	3 (39)	346	0.95	1	0.63	1
Item 1[a]			0.99	1	0.96	1

[a]Accuracy measures recalculated without Japanese responses.

The positive instances are the responses scored by humans as true and the negative instances are the ones scored by humans as false. The second row shows the results if we were to recalculate the accuracy measures above without considering the Japanese responses.

Results for Item 2

The following three tables of results correspond to Item 2.

Table 8

Item 2: False Versus True as Originally Automatically Scored

	Total	False	Non-null false	False distinct non-null	False distinct non-null uniform	True	True distinct	True distinct uniform
IE__EN	81	23	17	13	13	58	3	2
GB_EN	66	16	5	4	4	50	5	5
CA_EN	44	13	5	5	5	31	4	4
CL_ES	107	34	21	19	18	73	7	7
ES_ES	128	44	20	14	14	84	7	7
FR_FR	213	73	27	16	16	140	12	11
CA_FR	56	23	10	10	10	33	4	4
JP_JA	157	36	16	15	15	121	13	13

Training on IE_EN, a threshold of 0.69 is selected with one false response with a human label of false getting an updated score as true (with a 0.5 minimum *NDissimilarity* and a response of 5). To evaluate, the rest of the countries were used and results are listed in Table 10. In GB_EN, the two responses are corrected properly; after inspection, a 0.33 threshold is enough for the two responses. The response in CA_EN needs an *NDissimilarity* of 0.7. Hence, it gets missed with a 0.69 threshold. For ES_ES, the five responses pass the threshold. In fact, all responses require *NDissimilarity* of 0.2–0.55. Similarly, CL_ES's five responses pass with maximum *NDissimilarity* of 0.33. In particular, 0.13, 0.2, 0.27, 0.33 is the minimum proximity of each response to one of the true responses. For FR_FR, the eight responses pass with 0.06–0.56 *NDissimilarity* range. For CA_FR, two responses fail to pass the threshold requiring 0.74 and 0.75 *NDissimilarity*, while one response passes with 0.22 *NDissimilarity*. For JP_JA, five out of the five pass (0.01–0.17).

Table 9

Item 2: How Many False Get Corrected Properly to True?

	Non-null with original automatic score as false	Non-null original score false scored by humans as true	Non-null original score false with human score true with updated automatic score as true	Non-null original score false with human score false with updated automatic score as true
IE__EN_(training)	17	9	9	1
GB_EN	5	2	2	1
CA_EN	5	1	0	1
CL_ES	21	5	5	11
ES_ES	20	6	6	4
FR_FR	27	8	8	2
CA_FR	10	3	1	2
JP_JA	16	8	8	0

For this particular item, the majority of correct responses are the same. There is not much variety in correct responses with which to compare. Hence, a similar response to the following true response is deemed too distant, unfortunately:

> When you make a direct call to Portugal from another country, after dialing the number which gains access to the international service (which varies from country to country), you should dial 351 (the code for Portugal) and the inter-urban code without

On the other hand, a response in GB_EN whose original score is false and whose human score is false such as "For information about the codes for countries and places that are not included in the list, please dial: For help connecting a call, please dial: Some Country, Regional and City Codes In the case of countries marked with a "*" you only ne" gets a minimum *Dissimilarity* score of 0.673 with one true response, "351 (the code for Portugal) and the inter-urban code without the first 0."

In CA_EN, a response whose original score is false and whose human score is false such as "For information about the codes for countries and places that are not included in the list, please dial: 351" has a minimum *NDissimilarity* of 0.668 with one of the true responses*,*" you should dial 351 (the code for Portugal)." Hence, it passes the threshold of 0.69.

This latter example requires some knowledge about the context in the rest of the false response. We might want to consider the proximity or distance from more than one true response and/or other false responses, as we suggest in the clustering approach where all responses get compared.

For CL_ES, a minimum *NDissimilarity* is shared between two responses, 35 and 00 49 351. The original automatic scores for these two responses are false. They share the same *NDissimilarity* of 0.43 when compared to a correct response, 1351, while one of them is labeled true by humans and the other is labeled false.

For ES_ES, five responses that are supposed to be true all pass and their minimum *NDissimilarity* is either 0.2, 0.25, or 0.53, while the six false positives have *NDissimilarity* of 0.53, 0.55, 0.59, 0.67, 0.68.

For FR_FR, the two false positives pass with 0.59 and 0.57 *NDissimilarities*, while for CA_FR, the false positives pass with 0.54 and 0.62 *NDissimilarities*.

Table 10 summarizes the results.

Table 10

Item 2: How Many False Get Corrected Properly to True? (Summary)

Item	Training # of positive instances (# of negative instances)	Evaluation	Accuracy	Precision	Recall	True negative rate
Item 2	9 (8)	104	0.76	0.57	0.9	0.69
Item 2[a]			0.76	0.57	0.9	0.69

[a]Accuracy measures recalculated without Japanese responses. There would be no difference in the result if we were to exclude the Japanese responses.

Results for Item 3

The following three tables of results correspond to Item 3.

Table 11

Item 3: False Versus True as Originally Automatically Scored

	Total	False	Non-null false	False distinct non-null	False distinct non-null uniform	True	True distinct	True distinct uniform
IE_EN	81	20	11	10	10	61	1	1
GB_EN	63	17	8	7	7	46	4	4
CA_EN	46	13	4	4	4	33	3	3
CL_ES	101	38	9	8	8	63	2	2
ES_ES	115	42	12	9	9	73	3	3
FR_FR	216	71	19	15	15	145	5	5
CA_FR	39	11	1	1	1	28	3	3
JP_JA	148	24	6	6	6	124	4	4

For this item, except for Japanese where responses needed *NDissimilarity* of 0.6 and 0.2 to pass, most responses were corrected with the selected threshold of 0.09, based on the training set in IE_EN. For CL_ES, one response, "Agencia E ad y la Salud," that says something like "E Agency y and Health" for "European Agency for Safety and Health at Work" requires a 0.4 *NDissimilarity* to pass. For ES_ES, six out of eight responses pass with the 0.09 threshold. The other two require *NDissimilarity* of 0.17 and 0.13, respectively. For FR_FR, 10 out of the 12 pass, while the two remaining responses require 0.1 and 0.17 *NDissimilarity*.

Table 12

Item 3: How Many False Get Corrected Properly to True?

	Non-null with original automatic score as false	Non-null original score false scored by humans as true	Non-null original score false with human score true updated automatic score as true	Non-null original score false with human score false updated automatic score as true
IE_EN (training)	11	8	8	0
GB_EN	8	6	6	0
CA_EN	4	3	3	0
CL_ES	9	5	4	0
ES_ES	12	8	6	0
FR_FR	19	12	10	0
CA_FR	1	1	1	0
JP_JA	6	2	0	0

Table 13 summarizes the results.

Table 13

Item 3: How Many False Get Corrected Properly to True? (Summary)

Item	Training # of positive instances (# of negative instances)	Evaluation	Accuracy	Precision	Recall	True negative rate
Item 3	8(3)	60	0.88	1	0.81	1
Item 3[a]			0.94	1	0.86	1

[a]Accuracy measures recalculated without Japanese responses.

Results for Item 4

The following three tables of results correspond to Item 4.

Table 14

Item 4: False Versus True as Originally Automatically Scored

	Total	False	Non-null false	False distinct non-null	False distinct non-null uniform	True	True distinct	True distinct uniform
IE_EN	85	47	47	41	39	38	18	17
GB_EN	69	43	31	22	21	26	11	10
CA_EN	48	31	24	20	18	17	11	10
CL_ES	94	64	43	30	28	30	17	15
ES_ES	127	72	46	28	26	55	23	21
FR_FR	246	157	117	84	79	89	29	25
CA_FR	66	37	29	22	21	29	16	15
JP_JA	153	84	63	41	41	69	25	25

With no human-labeled true responses for IE_EN, we opted to use the GB_EN dataset and a threshold of 0.22 was selected. As mentioned earlier, there was no particular methodology on selecting the training dataset for this particular study. When there were no useful responses corresponding to a particular <country, language> pair, the next dataset available corresponding to any <country, language> pair was used to learn a threshold.

Table 15

Item 4: How Many False Get Corrected Properly to True?

	Non-null with original automatic score as false	Non-null original score false scored by humans as true	Non-null original score false with human score true updated automatic score as true	Non-null original score false with human score false with updated automatic score as true
IE_EN	47	0	0	0
GB_EN (training)	31	2	2	0
CA_EN	24	2	2	0
CL_ES	43	18	3	0
ES_ES	46	0	0	0
FR_FR	117	11	11	0
CA_FR	29	3	3	0
JP_JA	63	0	0	6

The two responses labeled true by humans in CA_EN each have a *Dissimilarity* of 0 with one of the true responses. It is similar for one response labeled true by humans in CA_FR. Hence, it is not clear why these have an original automatic score of false unless there is a space or some other grapheme missing that was not captured while extracting the text from the log file. For CL_ES, only three out of 12 pass while the rest have *NDissimilarity* ranging from 0.26–0.63.

Except for JP_JA, none of the false responses labeled by humans as false get an updated automatic score of true. The six responses in Japanese have minimum *NDissimilarity* ranging from 0.078–0.215.

Table 16 summarizes the results.

Table 16

Item 4: How Many False Get Corrected Properly to True? (Summary)

Item	Training # of positive instances (# of negative instances)	Evaluation	Accuracy	Precision	Recall	True negative rate
Item 4	2(29)	369	0.95	0.76	0.55	0.98
Item 4[a]			0.95	1	0.55	1

[a]Accuracy measures recalculated without Japanese responses. There would be no difference in the result if we were to exclude the Japanese responses.

For the four items in the four languages and the eight countries considered, there is a nontrivial percentage of responses whose original automatic score is false that get corrected properly to true, while each item has very different expected evidence and each human rater might have been more strict or lenient (no double scoring for the same responses and items have occurred to compare Human–Human agreement). However, as the precision and recall tables show, this is not the whole story. In one particular case (Item 2), the number of false positives is nontrivial. The questions then are: (a) Is erring on the positive side or the negative side better, that is, is it better to grant more true scores than false, and (b) Might it be the case that one can categorize the types of scoring rules associated with this item type into ones where this LCS method works better for some categories rather than others? Saying this, note that the number of training and evaluation data is very small and language-specific thresholds might hold better. For three of the four items, both Precision and Recall are either one or very close to one. Hence, these are considered excellent results. That said, evaluations with more items and responses will reveal the strengths and weaknesses of this approach.

Discussion

In the following, we describe some general observations and issues that might exist. These are divided into human scoring issues and issues with linguistic and cultural content analysis that have an effect on both human and automatic scoring.

Human Scoring

We found that human scoring was more challenging than expected. Deciding where one draws the line about what is considered false or true was not so straightforward, given the current content representation.

For instance, we looked at the following example:

> '「これは悪い知らせだ」とセクラーは言う。', '「2時間程度の短いトレーニングで、年配者のテスト結果が向上した。継続的にトレーニングを行えば、素晴らしい結果が出る」'

that, translated to English, is

> "'That's the bad news, says Sekular.'" A training session of as little as two hours improves the test results of older people. Sustained training can work wonders."

It was hard to decide whether to score this as true or false. It is true because the student responded correctly in the part about "A training session of as little …wonders." The additional part, that is, that violates the maximum, namely, "that's the bad news, says Sekular," appears in a very different column in the passage. Hence, it is unlikely that the student guessed the correct response. In fact, the prompt specifically asks for some other person's take on the issue. On the other hand, the additional part has no semantic relevance or coherence with the correct response. The human scoring allowed us to categorize responses according to the following:

Highlighting Beyond or off the Maximum

- What seems like a pure mechanical mistake: When a student seems to unintentionally omit or add and ends up with a response beyond the maximum.
- What seems semantically legitimate: When a student highlights additional information either as: (a) a specific detail such as an illustration or an example confirming the correct response, or, (b) what seems to be logical coherent

continuation of a correct response but not as specific as an illustration or an example.

- What is legitimately wrong: When a student highlights clearly wrong information—for instance, information that is incoherent with the maximum or highlighting the whole passage.

Highlighting Insufficient or Inadequate Minimum

- What seems like a pure mechanical mistake: Similar to the above case, a student omits or adds unintentionally and ends up with a response that is insufficient or inadequate.
- What seems semantically legitimate: When the occurrence of a certain minimum in a certain position is not acceptable, though it does not seem there is any contradictory or unsuitable context.
- What is legitimately wrong: When an item is clearly wrong, for instance, *stress* as a response to "how to train your memory?"

Question Marks

We give this nomenclature for cases where we are not sure why responses have been scored as such. For example, some responses that are clearly correct were given a false label automatically. This might have been due to missing spaces or preprocessing steps that were not extracted in our data.

A Deeper Content Analysis

We will present our observations in five categories.

Natural Language Translation-Based Issues

It might be possible that some meaning variations are introduced when translating from one language to another. In other words, students responding in Language *X* might be, unintentionally, favored over students responding in Language *Y*. The issues we noticed until now had to do with the selection of lexical entities by item developers. A lexical entity comprises a word, a compound, or a multiword lexical entity. In the rest of the document, we will just refer to it as a *word*.

Introducing ambiguities. This occurs when translating a passage from Language L1 to Language L2. A word that belongs to the minimum correct response is used in L2 in more than one position in the passage with the right context, while in L1 different words are used or the same word with different implicit contexts is used. This repetition might increase the ambiguity for students. For instance, in 集中的記憶訓練, the word *training* in Japanese is introduced in more than one position in the passage with the right context. Hence, either response is correct, while the rubrics specified one position to be correct and the other not. The reason is that in English there was no such reoccurrence or ambiguity between the two occurrences.

Reducing ambiguities. This occurs when translating a passage from Language L1 to Language L2. A word that belongs to the minimum correct response is used in L2 in more than one position in the passage that might reduce the ambiguity for students responding in L2. For instance, in English, the word *create* occurs more than once in a certain passage, yet only one occurrence of the word is associated with creativity and abstract artistic talents like poetry. The same happens in Spanish and French. When looking at the same passage in Japanese, two different words were used—one associated with abstract objects and one associated with concrete objects. A student responding in Japanese is more likely to select the word associated with abstract things when asked about art, while the students in other languages will not have that extra resolution of ambiguity between the occurrences of *create*.

Neutral but different translation. An example would be the use of numbers expressed in words rather than numerals, such as *twenty* versus *20*.

Content Word Order

Denoting a subject in a sentence by S, a verb by V, and an object by O, then languages could be divided between SVO, SOV, VSO, VOS, OVS, and OSV. We are used to languages with SVO orders, but actually the most common order in the languages of the world, including Japanese, is SOV. The other orders occur in very small percentages.

This might create issues because a content word such as a verb or a noun (object or a subject) might appear in the minimum. For example, consider the minimum evidence, “electronic products,” and a response, “Japanese build electronic products smoothly and elegantly.” In Japanese syntax, *electronic products* would not be placed between *build* and the adverbs, but located earlier, so Japanese students might highlight “build smoothly.” Hence, their response is correct, but it is inadequate minimum (because minimum evidence is "electronic products"). However, either the meaning is implicit or they missed "electronic products" by mistake.

Background, Cultural, or World Knowledge Variations

There are variations that we attribute to background, cultural, and world knowledge. A few examples follow.

The prefix used to place a phone call abroad varies from one country to another. Though this is stated in the passage corresponding to one item, it is an issue that people in general have when they move to a new country. In the United Kingdom, for instance, calling abroad requires 00, but in the United States, 011. Some students tend to highlight the 00 if there was any occurrence of 00 in the text, in addition to what is required based on, we hypothesize, the student's home country. The maximum in the scoring rules might have to take this into consideration.

Another instance is related to professions. The fact that architects are considered artists and not engineers in some countries might be a reason why many students highlight the fact that engineers are associated with architects in the text as a sign of being artistic. The same goes for the construction of structures such as bridges because it is creative and artistic.

Same Language, Different Variations

Within the same language, there were variations. Between the passage written in French in France and the one in Canada, there were many variations in the translation. For instance, *wonders* was translated to *wonders/merveilles* in France but translated to *miracles* in Canada. *Sustained* was kept *sustained* in Canadian French, but *regular/régulier* was used in France.

- French in France: Un entraînement régulier peut faire des merveilles.
- French in Canada: Un entraînement soutenu peut faire des miracles.

Many more variations existed. Consider another example in Spanish such as the following different translations or representations of the same text:

- Spanish in Chile: Sin embargo, la degeneración de la capacidad intelectual puede prevenirse de verdad, incluso aunque no nos dediquemos a ejercicios experimentales tan extremos. Como dice Lehrl, “cualquiera que esté interesado en cosas nuevas mantendrá la materia gris.”

- Spanish in Spain (Castellano): La degeneración de la capacidad intelectual de todos modos se puede evitar, incluso sin darse el lujo de considerar el pensamiento como un ejercicio experimental extremo. Tal como lo señala Lehrl: “cualquiera que esté interesado en nuevas cosas mantendrá la materia gris en forma.”

Learning the threshold using only one type of French or Spanish might not be as accurate for evaluation. Only a broader evaluation effort can tell.

Negation as Its Own Special Case

This case is for items where the minimum includes a negation. We only focus on explicit negations that are clearly modified by *not*. In our study we had no such items, but the approach we used will have to be updated to take care of, for instance, *not less than average* instead of *less than average* in the example in Table 1.

All of the above issues can be seen by looking at the data, but some can be inferred from the hierarchy of clusters (ranked in order). On the one hand, all types of issues apply to all languages, but that is not the only thing involved. In some cases, the students’ responses can be clustered as similar across languages. For instance, in one of the items, many students responding in English and Japanese selected the same wrong response:

> a training session of as little as two hours improves the test results of older people. Sustained training can work wonders. He is now pressing ahead with intensive memory exerices with a group of elders in good health who are intending to compete next year against young memory whiz kids.

The high-ranked clusters that we can perform over the proximity of responses to each other will confirm whether and how the rubrics can be refined.

Conclusion

We used one sequence alignment technique in order to evaluate multilingual student responses in international assessment. We showed it is an enhancement of what already exists, that is, the symbolic Boolean logic rules implementation.

In addition to automatic scoring, this solution plays an important role in clustering students' responses; consequently, it has a nontrivial impact on the refinement of the items and/or their scoring guidelines. The strengths of such an approach are that (a) it is language-independent, (b) it is adaptive, and, most importantly, (c) though designed for a specific problem, the proposed solution is general enough for any educational task that can be transformed into a sequential one even beyond text and speech applications. To name a few: It can be used for a set of actions expected from a student in simulations or learning trajectories as projected by a teacher, inside an intelligent tutoring system, or even in a game—or it can simply be used for a set of student clicks, button selections, or keyboard hits expected to reach a correct answer. This solution provides flexibility for existing automatic scoring techniques and potentially could provide more if coupled with statistical data mining techniques. The limitations of such an approach are that it considers the alphabet to be the graphemes without taking deeper linguistic features into account. Hence, a phenomenon like negation could be missed or needs a specific treatment by considering, for example, order and *n* consecutive graphemes as part of the alphabet.

The LCS approach was only one possible solution or similarity measure and within this approach; there were many choice points that could lead to various interesting research studies, such as the choice of the normalized dissimilarity, the approach of collapsing the row of the (dis)similarity matrix, the selection of a threshold, and even, probably, weighing different graphemes differently. Many choice points could be refined by additional data evaluation.

Our next steps will include trying several evaluation techniques, such as treating each language separately using bigger datasets and defining, comparing to a baseline, and considering additional attributes like the position of the characters in the passages. Also, this approach is being implemented operationally and an evaluation with more items and a larger number of responses will be available soon that will, empirically, help us in our measure selection. Furthermore, we need to conduct a comparative evaluation with a different alignment technique with various algorithms to find optimal alignments. These algorithms probably will be borrowed from the field of bioinformatics—in particular, how to select an optimal alignment. Finally, it is important to note that the size of datasets in our case, as compared to big data tasks that exist in bioinformatics or natural language processing, is very small.

References

Altschul, S. F., & Erickson, B. W. (1986). Optimal sequence alignment using affine gap costs. *Journal of Molecular Biology*, *48*, 603–616.

Altschul, S. F., Gish W., Miller W., Myers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. *Journal of Molecular Biology, 215*, 403–441

Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., & Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, *25*, 3389–3402.

Arratia, R., & Waterman, M. S. (1994). A phase transition for the score in matching random sequences allowing deletions. *Annals of Applied Probability*, *4*, 200–225.

Barzilay, R., & Lee, L. (2002, July). Bootstrapping lexical choice via multiple-sequence alignment. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, (EMNLP), Philadelphia, July 2002* (pp. 164–171). Stroudsburg, PA: Association for Computational Linguistics.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2001). *Introduction to algorithms* (2nd ed.). Cambridge, MA: MIT Press.

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors, *Communications of the ACM*, *7*, 171–176.

Gotoh, O. (1982). An improved algorithm for matching biological sequences. *Journal of Molecular Biology*, *162*, 705–708.

Hadlock, F. (1988). Minimum detour methods for string or sequence comparison. *Congressus Numerantium*, *61*, 263–274.

Hirschberg, D. S. (1975). A linear space algorithm for computing maximal common subsequences. *Communications of the ACM, 18,* 341–343.

Hirschberg, D. S. (1977). Algorithms for the longest common subsequence problem*. Journal of the ACM*, *24*(4), 664–675.

Huang, X., & Miller, W. (1991). A time-efficient, linear-space local similarity algorithm. *Advances in Applied Mathematics*, *12*, 337–357.

Hunt, J. W., & Szymanski, T. G. (1977). A fast algorithm for computing longest subsequences. *Communications of the ACM*, *20*(5), 350–353.

Iliopoulos, C. S., & Rahman, M. S. (2008). A new efficient algorithm for computing the longest common subsequence. *Theory of Computing Systems*, *45*(2), 355–371.

ISO 3166-2. (n.d.). In *Wikipedia*. Retrieved October 1, 2012, from http://en.wikipedia.org/wiki/ISO_3166-2

Kondrak, G. (2002). *Algorithms for language reconstruction*. Retrieved from http://webdocs.cs.ualberta.ca/~kondrak/papers/thesis.pdf

Levenshtein, V. I. (1965). Binary codes capable of correcting spurious insertions and deletions of ones. *Problems of Information Transmission*, *1*(1), 8–17.

Levenshtein, V. I. (1966) Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady 10*, 707–710.

List of ISO 639-1 codes. (n.d.). In *Wikipedia*. Retrieved October 1, 2012, from http://en.wikipedia.org/wiki/List_of_ISO_639-1_codes.

Lukashenko, R., Graudina, V., & Grundspenkis, J. (2007). Computer-based plagiarism detection methods and tools: An overview. *ACM International Conference on Computer Systems and Technologies*, *54*(3), 203–215.

Masek, W. J., & Paterson, M. S. (1980). A faster algorithm computing string edit distances. *Journal of Computer and System Sciences*, *20*(1), 18–31.

Myers, E. (1986). An O(ND) difference algorithm and its variations. *Algorithmica, 1,* 251–266.

Myers, E., & Miller, W. (1989). Optimal alignments in linear space. *Computer Applications in the Biosciences*, *4*(1), 11–17.

Nakatsu, N., Kambayashi, Y., & Yajima, S. (1982). A longest common subsequence algorithm suitable for similar text strings. *Acta Informatica*, *18*, 171–179.

Needleman, S. B., & Wunsch, C. D. (1970). Needleman-Wunsch algorithm for sequence similarity searches. *Journal of Molecular Biology*, *48*, 443–453.

Pearson, W. R., & Lipman, D. J. (1988). Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences of the United States of America*, *85*, 2444–2448.

Prinzie, A., & Van den Poel, D. (2005). Incorporating sequential information into traditional classification models by using an element/position-sensitive SAM. *Decision Support Systems*, *42*(2), 508–526.

Pucher, M., Türk, A., Ajmera, J., & Fecher, N. (2007, September). Phonetic distance measures for speech recognition vocabulary and grammar optimization. In *Proceedings of the 3rd Congress of the Alps Adria Acoustics Association*, Graz, Austria. Retrieved from http://www.alpsadriaacoustics.org/archives/Full%20Papers/Pucher_Phonetic%20Distance%20Measurement%20for%20Speech%20Recognition%20Vpcabulary%20and%20Grammar%20Optimization.pdf

Sasu, L. (2011). A probabilistic model for spelling correction. *Bulletin of the Transilvania University of Brasov*, *4*(53), 141–146.

Sellers, P. H. (1974). On the theory and computation of evolutionary distances. *SIAM Journal on Applied Mathematics*, *26*, 787–793.

Smith, T. F., & Waterman, M. S. (1981). Identification of common molecular subsequences. *Journal of Molecular Biology*, *147*, 195–197.

Su, Z., Ahn, B., Eom, K., Kang, M., Kim, J., & Kim, M. (2008, June). Plagiarism detection using Levenshtein distance and Smith-Waterman algorithm. In *Innovative Computing Information and Control (ICICIC), 2008 3rd International Conference* [CD-ROM]. New Brunswick, NJ: IEEE Press.

Thang, D. (2011). Algorithm to determine longest common subsequences of two finite languages. In N.-T. Nguyen, B. Trawinski, & J. J. Jung (Eds.), *Studies in computational intelligence: Vol. 351. New challenges for intelligent information and database systems,* (pp. 3–12). Berlin, Germany: Springer

Wagner, R. A., & Fischer, M. J. (1974). The string to string correction problem. *Journal of the ACM (JACM), 21*(1), 168–173.

Ziółko, B., Galka, J., Skurzok D., & Jadczyk, T. (2010, September). *Modified weighted Levenshtein distance in automatic speech recognition*. Paper presented at the 20th Economic Forum (Krajowa Konferencia), Krynica Zdrój, Poland.

Notes

[1] We will not make a distinction between character-based, alphabet-based, abjad-based, syllable-based, abugida-based languages, and so forth. We will just use *grapheme* to denote the smallest unit.

[2] Proofs showing the correctness of such algorithms are not included in this document.

[3] A programming language in continuous development since 1987; its main author is Jan Wielmaker.