Article

Educational and Psychological Measurement
2016, Vol. 76(2) 280–303
© The Author(s) 2015
Reprints and permissions:
sagepub.com/journalsPermissions.nav
DOI: 10.1177/0013164415590022
epm.sagepub.com
SAGE

Automatic Coding of Short Text Responses via Clustering in Educational Assessment

Fabian Zehner[1,2], Christine Sälzer[1,2], and Frank Goldhammer[2,3]

Abstract

Automatic coding of short text responses opens new doors in assessment. We implemented and integrated baseline methods of natural language processing and statistical modelling by means of software components that are available under open licenses. The accuracy of automatic text coding is demonstrated by using data collected in the *Programme for International Student Assessment* (PISA) 2012 in Germany. Free text responses of 10 items with $n = 41,990$ responses in total were analyzed. We further examined the effect of different methods, parameter values, and sample sizes on performance of the implemented system. The system reached fair to good up to excellent agreement with human codings ($.458 \leq \kappa \leq .959$). Especially items that are solved by naming specific semantic concepts appeared properly coded. The system performed equally well with $n \geq 1,661$ and somewhat poorer but still acceptable down to $n = 249$. Based on our findings, we discuss potential innovations for assessment that are enabled by automatic coding of short text responses.

Keywords

computer-based assessment, automatic coding, automatic short-answer grading, computer-automated scoring

[1]Technische Universität München, Munich, Germany
[2]Centre for International Student Assessment (ZIB) e.V., Munich, Frankfurt am Main, Kiel, Germany
[3]German Institute for International Educational Research, Frankfurt am Main, Germany

Corresponding Author:
Fabian Zehner, TUM School of Education, Centre for International Student Assessment (ZIB) e.V., Arcisstr. 21, 80331 Munich, Germany.
Email: fabian.zehner@tum.de

In automatic coding of short text responses (AC), a computer categorizes or scores short text responses. Such open-ended response formats, as opposed to closed ones, can improve an instrument's construct validity (e.g., for mathematics: cf. Birenbaum & Tatsuoka, 1987; Bridgeman,1991; for reading: cf. Millis, Magliano, Wiemer-Hastings, Todaro, & McNamara, 2011; Rauch & Hartig, 2010; Rupp, Ferne, & Choi, 2006). Yet today, closed formats have become state of the art, because they allow data to be easily processed and are more objective in nature. They do not require human coders, who entail some disadvantages (cf. Bejar, 2012): their subjective perspectives, varying abilities (e.g., coding experience and stamina), the need for consensus building measures, and, in turn, a high demand on resources. Taking a step back and reconsidering the response format's impact on construct validity, one realizes that these well-founded practical reasons in favor of closed formats might not be the most important decision criteria. Construct validity is the very base of all subsequent outcomes (e.g., analysis, conclusions, treatments) and, hence, can be claimed to be the most important criterion. Closed response formats should be used when the construct definition demands, or is supported by, closed response formats, and open-ended formats should be used when the construct definition requires open-ended ones.

One way to avoid or to compensate weaknesses of human coding is AC. Its employment has several implications. First, it is internally more consistent, because the final system is deterministic and always takes the same decision paths. This, in turn, circumvents consensus building measures. Second, the computer has stamina not to fatigue even at big data. Third, the coding of a new, unseen text response is done instantly and, thus, opens new doors in assessment. For example, computer adaptive testing (CAT) is constrained to the use of response formats that can be evaluated immediately. AC enlarges the scope of CAT to open-ended text responses. On the other hand, fourth, an AC system has its own limitations, errors, and costs (e.g., software development, manual coding for training data, etc.). The reader might notice that we use the term *coding* instead of the common *scoring*. That is because we regard scoring as a special case of coding and do not want to restrict AC's scope. In coding, a nominal category is assigned to an element; for example, a response could be categorized as either dealing with a financial or social aspect. On the other hand, in scoring, the categories additionally have a natural order; in the example, the social response could be favored over the financial one.

Since Carlson and Ward (1988), several research groups have been dealing with natural language processing (NLP) striving to automatically code short text responses. But in contrast to the strongly related field of essay grading, the respective methods are not commonly used in practice. Reasons for this, among others, may be proprietary licenses, under which most off-the-shelf software in this field is published, as well as a general human skepticism toward machines processing natural language (cf. Dennett, 1991).

In fact, a large number of open NLP libraries are available that can be combined with statistical components. In this article, we propose a collection of baseline methods and related software components that can be used under open licenses. We implemented and integrated these components and report analyses demonstrating the method collection's feasibility and accuracy by applying it to data assessed at the Programme for International Student Assessment (PISA) 2012 in Germany. PISA is initiated by the Organisation for Economic Cooperation and Development (OECD) and assesses scientific, mathematical, and reading literacy in a large-scale context, partly by administering open-ended items. Large-scale assessments, in particular, typically require enormous effort and resources in terms of human coding of open-ended responses. Naturally, they are a highly suitable field to apply AC. This seems particularly true for international studies, since they inherently endeavor to maximize consistency across different test languages (cf. OECD, 2013).

The three main constructs assessed in PISA exemplarily show the importance of incorporating open-ended response formats. First, PISA defines mathematical literacy as ''the capacity to formulate, employ, and interpret mathematics'' (OECD, 2013, p. 25); particularly the first two elements can hardly be validly assessed in closed formats. Second, scientific literacy is, among others, defined as ''an individual's scientific knowledge and use of that knowledge … [as well as their] awareness of how science and technology shape our material, intellectual and cultural environments'' (OECD, 2013, p. 100). Particularly the latter clearly shows the requirement for respondents to integrate individual knowledge, ideas, and values, which is best assessed in open-ended formats; otherwise, the given response options would crucially influence the respondents' cognition. This is also true for, third, the construct of reading literacy, defined as ''understanding, using, reflecting on and engaging with written texts … [which] requires some reflection, drawing on information from outside the text'' (OECD, 2013, p. 61).

It is beyond the scope of this article to provide an extensive overview about existing AC systems. Rather, we would like to refer to Burrows, Gurevych, and Stein (2014) who give a comprehensive overview. The main differences of the approach presented here and software opposed to existing systems are the following. On the one hand, the approach does not yet include the most sophisticated machine learning methods with the advantage of higher flexibility and transparency for researchers of the social sciences by, among others, applying clustering. On the other hand, the software will be made freely available with a graphical interface to encourage researchers to use AC.

Our study was led by three research questions that investigate, first, the performance of the AC system; second, the need of single steps in the proposed collection of methods as well as possible alternate configurations; and third, the required sample size for its employment. This article aims to demonstrate the performance of an AC system for short text responses that relies solely on baseline methods. Thereby, researchers shall be encouraged to design instruments with open-ended response format where appropriate and assessment practitioners to use them. The following sections present the collection of proposed methods for AC, how the empirical analysis of these methods was conducted, and the obtained results. The results illustrate the

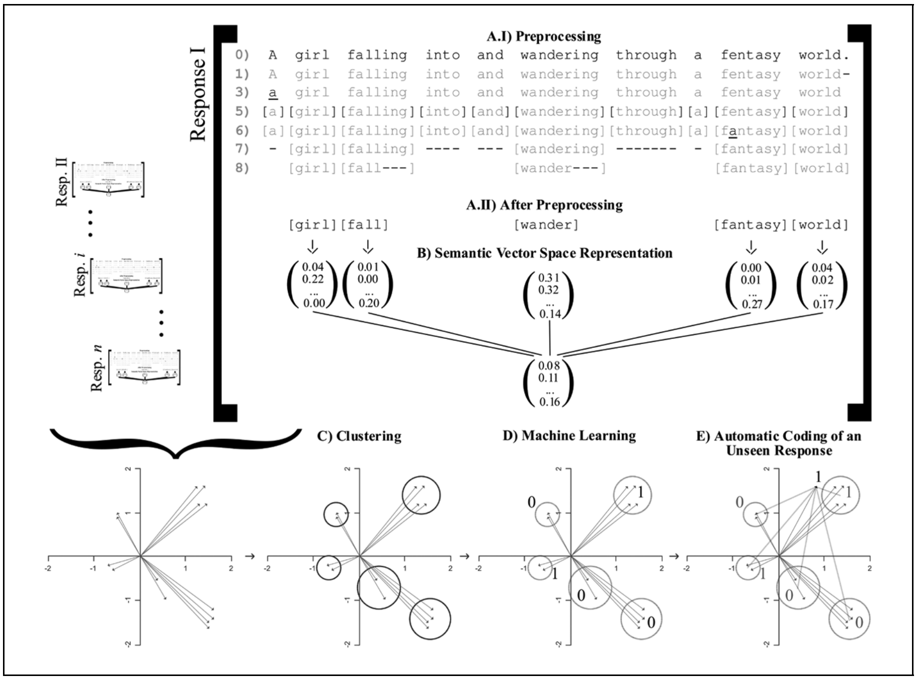
A.I) Preprocessing
0) A girl falling into and wandering through a fentasy world.
1) A girl falling into and wandering through a fentasy world-
3) a girl falling into and wandering through a fentasy world
5) [a][girl][falling][into][and][wandering][through][a][fentasy][world]
6) [a][girl][falling][into][and][wandering][through][a][fantasy][world]
7) - [girl][falling] ---- --- [wandering] ------- - [fantasy][world]
8) [girl][fall---] [wander---] [fantasy][world]
A.II) After Preprocessing
[girl][fall] [wander] [fantasy][world]
B) Semantic Vector Space Representation
Response I
Resp. II
Resp. i
Resp. n
C) Clustering
D) Machine Learning
E) Automatic Coding of an Unseen Response

Figure 1. Schematic figure of automatic coding (AC).

Note. First, the computer preprocesses the response (A) and extracts its semantics (B). These numerical semantic representations are used for clustering (C) and machine learning in which a code is assigned to each cluster (D). Finally, a new response receives the code of the cluster that is most similar to it (E).

overall performance of AC and how it depends on the item evoking the free text response, the selection of a particular method, and its configuration as well as sample size. The final discussion explicates exemplary ways in which AC could enhance educational assessment.

Proposed Collection of Methods for Automatic Coding

The following subsections describe methods that step-by-step process a test taker's free text response represented as a character string. First, NLP methods are used to transform each text response into a quantified representation of its semantics. Second, clustering methods are used to build a clustering model by grouping similar responses based on the numerical representations of the responses. Third, machine learning methods are applied to assign an interpretable code to each of these groups. For easier comprehensibility, an example response leads through the various steps: Let ‘‘A girl falling into and wandering through a fantasy world’’ be a test taker's response to an item asking what the main plot of Lewis Carroll's *Alice in Wonderland* is about. Figure 1 illustrates the entire procedure using this example response.

The method selection was led by two main paradigms that helpfully balance the simplicity of assumptions and their effectiveness in practice. First, the so-called *bag of words*–approach is chosen. This means, all terms given in a response are considered separately, irrespective of their order and references to each other. Second, the *co-occurrence* paradigm serves as a tool for automatic extraction of semantic relationships between words. The idea is to use the phenomenon that semantically similar terms occur in similar contexts—not necessarily in exactly the same context but at least indirectly in similar ones (cf. Landauer, McNamara, Dennis, & Kintsch, 2011).

Making Sense of Responses via NLP Techniques

In a first step, the computer preprocesses the response. Therefore, NLP techniques split the given text response into tokens (words) and transform these into more normalized ones, that is, their variation is reduced. The preprocessing steps are described in the following, and each step's effect on the example response is visualized in Part A of Figure 1.

(1) *Punctuation removal* omits punctuations, similar to (2) *digit removal* omitting digits. Next, (3) *decapitalization* converts all chars to lowercase. For some languages, language-specific techniques need to be added. For instance, in German (4) *umlaut-normalizing* is used to change umlauts into their corresponding vowel (e.g., *ä* into *a*). Splitting the response into tokens, meant to be the level of analysis, is called (5) *tokenizing*. Next, it is advisable to apply (6) *spelling correction*. Then, functional words are omitted that do not carry crucial semantic information themselves. This is called (7) *stop word removal* and typically applies to words such as articles and conjunctions. As last very important variation reduction, (8) *stemming* is applied, which means to cut off affixes. The example now reads: *girl, fall, wander, fantasy, world.*

These preprocessing techniques are typically useful. Nevertheless, in some cases, it might be reasonable to adapt their assembly. If, for example, digits are important in the item's solution, they, of course, should not be omitted but considered. Furthermore, it is possible to replace stemming by the more sophisticated *lemmatization* (using the basic word form instead of just cutting off affixes), and dependent on language and application, it can be useful to apply *compound splitting* (splitting words that are made of more than one stem). The above listed eight techniques have been used for analysis presented in this article.

Now, having reduced linguistic variation in responses, the computer needs some kind of numerical representation of the remaining tokens' semantics (ignoring order information as a *bag of words*–approach is used). Therefore, a big text corpus is automatically analyzed by a statistical method—using the co-occurrence paradigm—to build the machine's lexical knowledge. This is often called *semantic space*. Wikipedia, for instance, can serve as an adequate corpus. If feasible, multiple sources can lead to higher performance (Szarvas, Zesch, & Gurevych, 2011). The semantic spaces used for analysis reported in this article were built on basis of a German Wikipedia dump[1] (from June 13, 2013) as German text responses were to be coded.

Solely in cases of items that only evoke a strongly limited amount of different words in the responses—for example, if test takers are required to repeat four terms explicitly given in the stimulus—it can be conceivable to disregard the words' semantics. In such cases, it is also possible to test the plain existence of words and to continue with clustering (see next subsection). Nevertheless, in most cases considering the semantics is worth the effort, and in the empirical part, we investigate whether the usage of plain words can outperform the usage of semantics at all.

A common way to model semantics is through *vector space models*. These are hyperdimensional spaces in which each term (e.g., *fantasy*) is represented by a vector. Similar vectors (terms) are semantically similar (e.g., *fantasy* and *imagination*), defined as two vectors having a small angle, or more precisely as a high cosine of two vectors. Such semantic spaces can be computed by, for instance, *Latent Semantic Analysis* (LSA; Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990) or *Explicit Semantic Analysis* (ESA; Gabrilovich & Markovitch, 2007). Both start with a co-occurrence matrix, called ''term × document-matrix,'' carrying frequencies of how often which term occurs in which context (i.e., document, paragraph, or sentence). This matrix is weighted based on the idea of relevance feedback (Salton & Buckley, 1990), giving more weight to terms occurring in rather few, specific contexts over those occurring diffusely across many contexts. This way, words that adhere to a specific semantic context—which typically is rather true for autosemantic opposed to function words—become important carriers of this context's semantic information. Typically, relevance feedback is applied using the log-Entropy function (Dumais, 1991). Creating a semantic space by means of ESA includes several steps (for details, see Gabrilovich & Markovitch, 2009) to finally gain a vector model describing the semantic space. The ESA is based on a manually structured text corpus, that is, documents (such as articles in Wikipedia) are included as intact entities. Within this natural structure (e.g., composed of articles) each element corresponds to one dimension in space, often leading to a very high number of dimensions. For instance, a term such as *fantasy* is represented by a vector comprising over one million (total number of articles) weights, each indicating the relatedness between an article and the term. The second approach to create a semantic space is LSA, which performs a singular value decomposition on the weighted term × document-matrix. Then, as text corpora are samples of language containing noise, and as contexts are assumed to be correlated, the decomposed matrix is reduced to less—often 300—dimensions, comparable with factor analysis.[2] For example, a term such as *fantasy* is represented by a vector comprising 300 values, each indicating the relatedness between a latent semantic concept and the term. For details of LSA, see Landauer (2011) and Martin and Berry (2011).

An important matter in LSA is domain specificity of text corpora. When taking Wikipedia as a basis, one can make use of its internal, explicitly connected structure. One way to gain a domain-specific corpus is to start at an article that is strongly related to the item's topic (e.g., *Alice's Adventures in Wonderland*) and crawl the incoming and outgoing links, also called ''children,'' to related articles (e.g., *Fantasy*). Dependent on the desired corpus size and the initial article's connectedness, it is reasonable to do this within a degree of connectedness of 1 to 3 (e.g., 2 meaning to take articles' children as well as children's children into the corpus). This procedure can easily be adapted. For example, more than one starting point can be chosen to gather bigger corpora, or filters can be applied as to which kind of articles must not be included (e.g., articles only dealing with a specific date; for more such considerations, see Gabrilovich & Markovitch, 2009).

Having provided the computer with semantic knowledge, the semantics of the responses can be extracted by computing the centroid vector of all tokens in the preprocessed response. At this stage, the example response comprises five tokens (cf. Part B in Figure 1). Each is represented by, assuming LSA was used, a 300-dimensional vector. Finally, the five vectors' centroid is computed—that is one 300-dimensional vector constituting the example response's total, or average, semantics. In all further analyses this centroid vector is the response's representation.

Model Building via Clustering

Represented by their semantic centroid vector, all responses given in the data are spread across the semantic space. If an appropriate semantic space has been built, the responses now typically form groups (cf. Part C in Figure 1). This means that some responses are more semantically similar to each other than to others—to put it more precisely, responses in one group contain semantically similar words. This perspective on the data can easily be applied by conducting a cluster analysis. In case the data are meant to be explored only, all responses are taken into account for model building. Otherwise, if some sort of supervised learning takes place—which is going to be explained in the next subsection—only a subset of responses (training data) is considered for this model building step.

Here, an agglomerative hierarchical cluster analysis is used that comprises three steps.[3] First, the distance between every pair of responses is computed. Second, every response is regarded as being its own cluster at the beginning. An agglomeration method iteratively decides which two clusters are minimal distant from each other and are to be merged to one cluster in the next iteration step. The agglomeration stops when all responses are assigned to one single cluster. Third, the researcher needs to decide which number of clusters is the best solution. This is an empirical matter and can be determined by plotting the distances (''rest''-component) of clusters for each solution and applying the elbow criterion. The respective number of clusters constitutes the desired solution at which the rest-component decreases significantly higher, relative to solutions with more clusters. See Rasch, Kubinger, and Yanagida (2011) for a concise description how this is done using R or SPSS.

A cluster analysis has several adjustable parameters. One important decision is how to define similarity, which can either be done by Euclidean distance or—worthwhile although computationally more expensive—cosine. The latter means to divide the vectors' dot product (Martin & Berry, 2011) by the product of their vector lengths to apply an L2 norm, which is done to neutralize different document lengths (Gabrilovich & Markovitch, 2009). Cosine's advantage over Euclidean distance is to take only the vector directions into account—and not their lengths—because in the semantic space a vector's length mainly depends on the term's frequency, and as a term's frequency does not carry semantic information, the distance metric should incorporate only the vector's direction. Clustering algorithms base on *dis*similarity, and hence, the cosine's inverse, called *arccosine*, should be used. Besides choosing a measure of dissimilarity, the researcher needs to decide on the number of clusters as well as on an agglomeration method, such as Ward's method, which merges those two clusters leading to the smallest increase of variance within clusters (Ward, 1963). Nevertheless, all these parameters' optima are susceptible to the data's nature and, therefore, can be adapted with respect to theoretical or empirical knowledge about the respective item.

Up to this point, it is possible to work with text response data only. This is reasonable for data exploration and is called unsupervised learning; no further criterion is available (e.g., a variable indicating whether a response is *correct* or *incorrect*) that would allow model optimization with regard to this criterion. For example, the researcher can choose the number of clusters in an unsupervised manner as described above. But if the overall aim is to do some kind of supervised learning (e.g., scoring responses), it is reasonable to vary the number of clusters and choose the best performing one in terms of human–computer agreement. The following subsection about supervised learning describes how an interpretation such as *correct* or *incorrect* can be assigned to every cluster.

Assigning Codes to Neutral Categories via Supervised Machine Learning

Where the overall goal is AC, meaning that one of a fixed range of values—called *class*—needs to be assigned to a text response (e.g., *correct*), an external criterion is used by which to learn the relationship between the semantic vectors and the intended code. This kind of procedure is called supervised learning. Such an external criterion can be judgments by trained human coders, also called classification. Ideally, the whole data set is classified.

Supervised machine learning procedures are separated into two phases: *training* and *test*. In each phase only a subset of data is used, while the rest is put aside to control overfitting. Overfitting is given if a model is optimized too thoroughly, because it then performs very well on the data it was trained on but terribly poor on unseen data; while to apply models on unseen data usually is the purpose of building them. Hence, the data is split into 10 pieces, called *folds*. Nine folds are used together at a time to train the model while its performance is tested on the remaining 10th fold afterward. This is repeated 10 times so that every fold serves as test data once. Yet two things need to be considered. First, the data's overall class distribution should be represented in each fold; this is called *stratification*. Second, because still then chance highly impacts performance while choosing the folds, the whole procedure is again repeated

10 times, resulting in 100 performance estimates. This is called *stratified, repeated 10-fold cross-validation*; details can be found at Witten, Frank, and Hall (2011), and for comparison with other methods, see Borra and Di Ciaccio (2010).

In the training phase, clusters are built as described in the previous subsection. Then, to each cluster one code is assigned to (cf. Part D in Figure 1), determined by the within-cluster class distribution. For example, while scoring, a cluster might mainly comprise responses labelled as *correct*; hence, the cluster code is: *correct*. The decision which code is assigned to a cluster is based on the highest conditional probability using Bayes theorem,

$$P(c_i|g_j)_{k,j} = \frac{P(g_j|c_i)^* P(c_i)}{P(g_j)},$$

representing the probability that response k, belonging to the jth cluster g, has been assigned to the ith class c by the external criterion (e.g., human coder). For example, the probability of a response that is known to be member of Cluster 13, which is true for $\hat{P}(g_j) = 25\%$ of all responses, that it had been assigned to the class *correct* by the human coder, which is true for $\hat{P}(c_i) = 76\%$ of all responses, equals 96% if $\hat{P}(g_j|c_i) = 31\%$ of all correct responses belong to Cluster 13.

In the test phase, the unseen responses are classified by using the cluster model and the cluster codes. At first, the highest similarity, for instance, in terms of cosine, between the unseen response and a cluster centroid determines to which cluster the response is assigned to. Then, this cluster's code is assumed to be the response's code (cf. Part E in Figure 1).

At the end of the day, the automatic classifier needs to be evaluated. Its performance is often called *accuracy*. The simplest accuracy coefficient is percentage of agreement between computer and human. Another important one is kappa, which corrects for skewed class distribution.

Method

Measures of Coder Agreement

The Results section reports percentages of human–computer agreement ($\%_{h:c}$) as well as Cohen's kappa ($\kappa_{h:c}$) for data from PISA 2012. These statistics were computed by averaging the results from repeated 10-fold cross-validation. The kappa coefficient was transformed prior to averaging accordingly to Fisher (1915, 1921) and retransformed for reporting, on which the definition of Fleiss (1981) is applied: Values of $.40 \leq \kappa \leq .75$ constitute *fair to good*, lower values *poor*, and higher ones *excellent agreement beyond chance*. Moreover, a corrected coefficient of percentage of agreement,

$$\lambda_{h:c} = \frac{\%_{h:c_i} - \%_{h:c_1}}{100 - \%_{h:c_1}},$$

is reported, giving the proportion of the actual human–computer agreement's increase to the highest attainable increase; therefore, the percentage of agreement for the model with only one cluster ($\%_{h:c_1}$) is subtracted from the percentage of agreement for the final model with *i* clusters ($\%_{h:c_i}$), and this difference is divided by 100 minus the percentage of agreement for the model with one cluster, constituting the highest attainable increase. Unlike Kappa, this coefficient does not only correct for agreement by chance based on the class distribution but also for the trivial coding of empty responses. It further indicates how clustering affects performance in the data. Finally, Fleiss' kappa for multiple raters is used to report the system's internal reliability across repetitions ($\kappa_{c:c}$).

Materials

Items in PISA typically comprise a stimulus and a question referring to it. Responses used for analysis stem from 10 dichotomous items; eight items assessing reading literacy as well as one assessing scientific and mathematical literacy each. That is, there are items of three different constructs and specific coding guides for human coders for every item guiding them to code responses as either *correct* or *incorrect*. Although only items with dichotomous coding were used in this study, constituting the vast majority in PISA, the chosen approach to AC is also applicable to polytomously coded items. Because of repeated measurements, partly using the same items across cycles, the item contents are confidential and cannot be described here. The 10 items are listed in Table 1, each given a name for better traceability in the following. Prior to item selection, a theoretical framework had been developed as to which item characteristics might potentially influence AC performance using the proposed methods. According to this scheme, the selected items were intended to vary heterogeneously in their characteristics to test the AC method's scope.

As presented in Table 1, the items ranged from difficult (10%) to easy (83%) with the majority being medium difficult and a slightly skewed distribution toward easiness. For the reading items, the assessed aspect according to the PISA framework (OECD, 2013) mainly varied between the two of three aspects *Integrate and Interpret* and *Reflect and Evaluate* that both typically evoke more complex answers in linguistic terms than the third aspect does. The third aspect, *Access and Retrieve*, was only represented by one item and is assigned to items asking the test taker to find the relevant information given in the stimulus and repeat it, resulting in low language diversity in the responses.

Analyses

According to the first research question, we report the measures of coder agreement introduced above for all items to show the AC performance. For each item, we varied the number of clusters from 1 to 1,000 and report the best performing solution using cosine and Ward's method for clustering. We cross-validated all measures by

Table 1. Item Characteristics.

Item	Domain[a]	Aspect[b]	Correct	*n*	Words[c]
1. Explain protagonist's feeling	read	B	83%	4,152	12.3 (4.6)
2. Evaluate statement	read	C	43%	4,234	15.6 (9.0)
3. Interpret the author's intention	read	B	10%	4,234	12.5 (6.3)
4. List recall	read	A	59%	4,223	5.6 (3.0)
5. Evaluate stylistic element	read	C	56%	4,234	14.7 (6.2)
6. Verbal production	read	B	80%	4,152	12.4 (6.9)
7. Select and judge	read	C	68%	4,152	13.6 (7.0)
8. Explain story element	read	B	69%	4,223	14.4 (5.5)
9. Math	math	M	35%	4,205	14.0 (6.8)
10. Science	scie	S	58%	4,181	11.1 (5.2)
Total			56%	41,990	12.6 (6.1)

[a]One of *read*ing, *math*ematics, *scie*nce. [b]According to PISA framework (OECD, 2013), A = Access & Retrieve; B = Integrate & Interpret; C = Reflect & Evaluate; M = Uncertainty & Data; S = Explain Phenomena Scientifically. [c]Word count in nonempty responses on average (with *SD*).

stratified, repeated (10 times) 10-fold cross-validation, which is also true for the analyses described next.

Following the second research question, the relationship between performance and different methods or their configurations were examined. Therefore, seven analyses were carried out and are reported for single representative items. In Analysis I, the need for semantic space building opposed to the use of plain words is demonstrated. Analysis II illuminates the impact of spelling correction on AC performance by comparing uncorrected, automatically corrected, and manually corrected responses. The latter was possible by means of the transcription process (cf. next subsection). Semantic space building was varied in Analysis III using ESA or LSA, in Analysis IV by using different text corpora, and in Analysis V by varying the number of LSA dimensions. In Analysis VI, various common distance metrics were applied (*cosine*, *Euclidean*, *Chebyshev*, *Manhattan*, *Canberra*), and in Analysis VII, we studied the impact of different agglomeration methods (*Ward's*[4] and *McQuitty's method*; *Single*, *Complete*, *Average*, *Median*, and *Centroid Linkage*). For each variation, all other parameters and methods were set constant to those from the analyses for the first research question despite the number of clusters that was set with respect to the performance optimum. Percentages of agreement served as dependent variable in these experiments. The runs within one experiment were compared by considering the difference of means and the dispersions, for which percentage of agreement is the optimal measure.

Finally, in the third research question, we went about the relationship between performance and sample size, investigating to which extent the methods were applicable to studies with less data by conducting a simulation. To reduce random errors, we designed the simulation similar to repeated 10-fold cross-validation, in that, we first split the data into 10 stratified parts (called metafolds in the following) and subsequently excluded one metafold at a time until only one was left. At each of these reduction steps, all left metafolds were merged and an independent stratified, repeated (10 times) 10-fold cross-validation was run on them, disregarding the previous metafold assignments. For example, after the first reduction $n = 4{,}152 - 415 = 3{,}737$ responses contributed to the analysis, in the next step only $n = 3{,}737 - 415 = 3{,}322$ did. Next, when only 415 responses were left, the last remaining metafold was split into another 10 smaller metafolds to gain more fine-grained reduction steps within the last 10th; and again, we subsequently excluded these smaller metafolds until only one was left and ran independent repeated 10-fold cross-validations on the remaining data at each step. This procedure was repeated 10 times, each time shifting the order of metafold exclusion to reduce the impact of the metafold sampling itself. Figure 2 visualizes this design. The simulation used the data of Item 7, *Select and judge*, which turned out to be representative regarding the obtained results in various analyses and was automatically coded medium well—with that, being sensible to improvements and worsening caused by sample size variation. Again, all parameters and methods were set constant to those from the analyses for the first research question despite the number of clusters that was set with respect to the performance optimum.

Participants and Procedure

The responses we analyzed come from the German PISA 2012 sample. This includes a representative sample of 15-year-old students as well as a representative sample of ninth graders in Germany. A detailed sample description can be found at Prenzel, Sälzer, Klieme, and Köller (2013) and OECD (2014). Due to a booklet design, the numbers of test takers varied for each item ($4{,}152 \geq n \geq 4{,}234$; cf. Table 1).

In PISA 2012, *reading*, *maths*, and *science* were assessed paper based. Hence, booklets needed to be scanned and responses were transcribed by six persons. To reduce typos, an additional mechanism was employed. Misspelled terms were annotated with their correct spelling, which additionally served as upper bound of achievable improvement by spelling correction in Analysis II (cf. subsection *Relationship Between Performance and Alternation of Parameter Values or Methods*). Only for the additional typo reduction mechanism, the misspelled terms were replaced by their annotation and these manually corrected data were put into Microsoft Word 2013 and checked for left misspellings (that now could only stem from transcription typos) using spelling correction. In total, 0.2% of all words were revealed to contain a typo this way, which were corrected in the data. In the analyses, the original responses were used disregarding the manual correction but applying an automatic spelling correction on students' misspellings.

Software

The software programmed for the aforementioned procedure implements open software, libraries, and packages. First, a database storing the downloaded Wikipedia

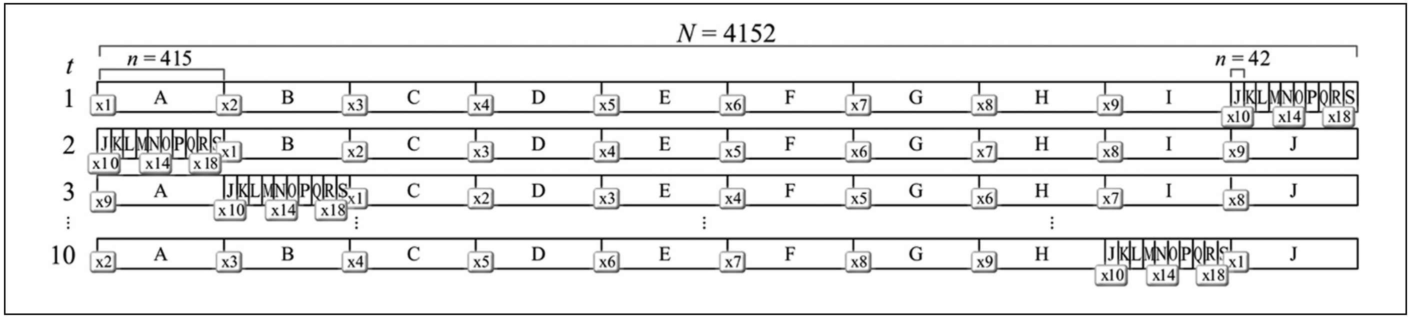

Figure 2. Sample size simulation design.

Note. The *x*-labels indicate which metafold was excluded in which order; for example, in the second repetition (t = 2), the data in Metafold E is included in the first three runs and excluded for all subsequent ones (×4) of this repetition. See the text for a more detailed description.

Table 2. Accuracy and Reliability.

Item	$\%_{h:c}$ [a]	$\lambda_{h:c}$ [b]	$\kappa_{h:c}$ [c]	$\kappa_{c:c}$ [d]
1. Explain protagonist's feeling	91.2[±0.2]	16.5	.533[.519; .547]	.814
2. Evaluate statement	86.5[±0.3]	63.8	.729[.723; .735]	.910
3. Interpret the author's intention	90.4[±0.2]	9.6	.458[.444; .472]	.738
4. List recall	98.4[±0.1]	84.2	.955[.952; .959]	.983
5. Evaluate stylistic element	76.2[±0.4]	12.4	.503[.495; .511]	.771
6. Verbal production	92.7[±0.2]	39.7	.704[.695; .713]	.925
7. Select and judge	86.6[±0.3]	39.2	.679[.672; .686]	.875
8. Explain story element	86.7[±0.3]	41.5	.676[.668; .683]	.886
9. Math	85.0[±0.3]	56.6	.669[.661; .676]	.822
10. Science	88.4[±0.3]	45.4	.761[.754; .767]	.918

Note. 95% confidence intervals given in brackets. [a]Percentage of human–computer agreement. [b]Relative accuracy increase by clustering; $\lambda_{h:c} = \frac{\%_{h:c_i} - \%_{h:c_1}}{100 - \%_{h:c_1}}$ (for details cf. Methods section). [c]Cohen's kappa for human–computer agreement. [d]Fleiss' kappa for within-computer reliability across repetitions.

dump (cf. subsection *Making Sense of Responses via NLP Techniques*) is built by using JWPL (Zesch, Müller, & Gurevych, 2008), which also comes with an application programming interface to access the corpus data. DKPro Similarity (Bär, Zesch, & Gurevych, 2013), which in turn primarily utilizes S-Space (Jurgens & Stevens, 2010), is used to build a vector space model. The response processing makes use of components offered in DKPro Core (Gurevych et al., 2007), which fit into the Apache UIMA Framework (Ferrucci & Lally, 2004). For stemming, Snowball (Porter, 2001) is used. For statistical matters, such as clustering, the software evokes R (R Core Team, 2014).

Results

Overall Performance and Its Relationship to Item Characteristics

As indicated in Table 2, the AC can lead to good up to excellent human–computer agreement; still, its performance varied by item remarkably. Across all items, the system reached high percentages of agreement from 76% to 98%, whereas the kappa values ranged from fair to good—Item 3, *Interpret the author's intention* ($\kappa_{h:c} = .458$)—up to excellent agreement beyond chance—Items 10, *Science* ($\kappa_{h:c} = .761$), and Item 4, *List recall* ($\kappa_{h:c} = .955$). Besides Item 3, Item 1, *Explain protagonist's feeling* ($\kappa_{h:c} = .533$), and Item 5, *Evaluate stylistic-element* ($\kappa_{h:c} = .503$), attained the poorest agreement beyond chance, and the five other items reached good agreement beyond chance ($.669 \leq \kappa_{h:c} \leq .729$). Whereas the kappa coefficient identifies the just named three items as performing poorest but still to a fair to good extent, the coefficient $\lambda_{h:c}$ underlines that their AC performance did not crucially improve by clustering ($9.6 \leq \lambda_{h:c} \leq 16.5$). A detailed analysis showed that the responses' correctness to items 3 and 5 could partially be affected by other linguistic elements than pure semantics, while the reasons for items' rather poor AC were not obvious. On the other hand, especially for Item 4, *List recall*, the human–computer agreement was excellent and it increased remarkably by clustering ($\lambda_{h:c} = 84.2$) as it also did for Item 2, *Evaluate statement* ($\lambda_{h:c} = 63.8$), and Item 9, *Math* ($\lambda_{h:c} = 56.6$). These items share one important characteristic, namely, the evoked response's correctness is determined by the test taker expressing specific semantic concepts. In Item 4, *List recall*, this is done in a very simple way by just listing terms while in the other items the verbal constructions need to be more complex to be able to express the required semantics. Finally, the system's reliability mainly attained excellent agreement with $.771 \leq \kappa_{c:c} \leq .983$, except of the AC of the already discussed Item 3 ($\kappa_{c:c} = .738$).

Relationship Between Performance and Alternation of Parameter Values or Methods

Competing methods and parameter values were compared in seven analyses with regard to percentage of human–computer agreement ($\%_{h:c}$) whereas the sets of 100 percentage values from stratified, repeated 10-fold cross-validation served as sample for the respective experimental conditions. Most analyses were conducted using the data of Item 7, *Select and judge*, constituting a medium well working item for AC. This way, the measure of agreement could optimally be affected by the experimental variation, which would not have been the case for an item performing perfectly or not at all. Only in the first analysis, the data of another item were used.

To take a conservative approach, we based Analysis I on Item 4, *List recall*, which asked test takers to simply name four terms that are explicitly given in the stimulus text. In such a case, one could assume that testing for the bare presence of terms might be sufficient. The analysis showed that using a semantic space opposed to simply using the presence of words results in significantly better AC, $t(197) = 3.34, p < .001, d = 0.94 (\hat{=}\ 0.3\%)$—although with small effect due to the conservative approach. Whereas the semantic space needed 65 clusters to reach its optimal performance, the usage of raw words already required 500 clusters for this simple item to do so.

Analysis II studied the impact of spelling correction on performance comparing three conditions: no, automatic, and manual spelling correction. The latter constituted an upper bound for the possible effect of spelling correction on performance of how much an almost perfect spelling correction could improve the AC. Using an ANOVA, the analysis revealed that the automatic spelling correction neither improved performance significantly opposed to dispensing with it nor did the performance with manual spelling correction differ significantly, $F(297, 2) = 2.1, p = .130$. To further investigate if this finding can be generalized across items for the PISA data, cross-checks on other items were run. These showed the necessity of spelling correction depends on the item's nature—the same analysis with the data of Item 10, *Science*, yielded in similarly insignificant results, $F(297, 2) = 2.4, p = .089$, whereas performance with and without spelling correction for Item 6, *Verbal production*, differed significantly with relevant effect size, $t(197) = 5.81, p < .001, d = 0.82(\hat{=} 0.9\%)$.

In Analysis III, the semantic space building was varied between the two methods ESA and LSA. Results showed that ESA performs poorer in comparison with LSA, $t(196) = 3.32, p = .001, d = 0.47(\hat{=} 0.8\%)$.

Analysis IV varied the text corpus on which the semantic space was based on. Five different text corpora were compared of which three closely dealt with the item contents and two did not. Additionally, they varied in size. Results showed that the selection of a text corpus affects AC-performance significantly, according to a one-way ANOVA, $F(4, 495) = 580.20, p < .001$, and a Newman Keuls procedure (SNK) revealed significant differences between every pair of corpora except for one pair that was equal in size but with one corpus having contents close to the item contents and the other one not, $SNK = -0.4\%, p = .094$. However, another corpus, which also closely dealt with the item contents but was bigger in size, gained a significantly higher performance level, $SNK = 0.9\%, p < .001$. The smallest corpus, not dealing with the item contents, performed poorest, $SNK = -7.7\%, p < .001$ (compared with the next better performing one). Disregarding this exceptionally poor performing corpus, the performances differed in a range of 1.8%. Thus, primarily, corpus size seems to matter, and content similarity only matters secondarily, as long as the corpus is big enough still to deal with the most important terms of the item contents.

Analysis V compared different numbers of extracted dimensions in LSA (50, 100, 200, 300, 400, 500, and 550). Results of an overall significant ANOVA, $F(6, 693) = 4.48, p < .001$, indicated that the number of dimensions only matter inasmuch as they should not decline below 100. The only significant differences stemmed from comparisons with 50 dimensions, $SNK = -0.5\%, p = .03$ (compared with next better performing one).

In Analysis VI, we studied the impact of different clustering distance metrics on performance. Results demonstrated significant differences between distance metrics in an ANOVA, which were also remarkable in effect size, $F(4, 495) = 121.30$, $p < .001$. Three groups crystallized, in that cosine, Euclidean distance, and Manhattan distance performed equally well while Chebyshev performed significantly poorer, $SNK = -2.2\%, p < .001$ (compared with Euclidean distance), and Canberra performed worse still, $SNK = -1.0\%, p < .001$ (compared with Chebyshev).

Finally in Analysis VII different agglomeration methods for clustering were compared. A similar pattern like in the previous analysis showed up as the agglomeration methods also formed three groups of performance levels with significant deviation in an ANOVA, $F(7, 792) = 125.60, p < .001$. No difference in performance could be found between Ward's method, Ward.D2, Complete Linkage, and McQuitty's method, $SNK = -0.5\%, p = .116$ (comparing the best with the poorest performer in this group). Another group was formed by Single, Median, and Centroid linkage that performed poorest. In between these two groups, Average linkage performed intermediately well; $SNK = -1.3\%, p < .001$ (compared with the poorest method of the well performing group), $SNK = 2.1\%, p < .001$ (compared with the best method of the poorly performing group).

Relationship Between Performance and Sample Size

In the sample size simulation, sample sizes were varied to study the performance's dependency on sample size. As results showed, AC performance was affected by sample size both significantly and relevant in effect size—$F(18,18081) = 82.1, p < .001$ (ANOVA); $SNK = 6.3\%, p < .001$, comparing the best with the poorest performance ($n = 3,322$ and 84). Applying an SNK procedure, five homogeneous groups with respect to performance were built. It appeared that the proposed AC methods can be applied with three different sample size ranges (I) to (III) down to $n = 249$ that result in slightly different but acceptable performance levels (cf. Figure 3). As illustrated in the figure, performance started to decline crucially with a smaller sample size than 249. In the first range of $n = [1661,\ 4152]$ (I), AC performed best and without significant within-differences. Second, $n = 1,246$ (II) performed significantly poorer compared to the second poorest performance of the top performers, $SNK = -1.1\%, p = .009$. Third, opposed to this intermediately performing $n = 1,246$, a last range of sample sizes (III) with reduced but acceptable performance comprised the range of $n = [249, 831]$, $SNK = -0.9\%, p = .022$. Still further reducing the sample size to $n = 207$ (IV) again led to a significant decline, $SNK = 0.9\%, p = .015$, and finally, the range of poorest performance with $n = [42, 166]$ (V) lost another $SNK = 0.8\%$, $p = .040$. While for Range I clustering highly improved AC performance ($\lambda_{h:c} = 29.2$ for poorest performance), it still did so to a lesser extent for Range III ($\lambda_{h:c} = 13.7$ for poorest performance), but the performance increase by clustering was not significantly different from 0 at all when using data with $n \leq 125$.

Discussion

In terms of agreement with human coders, the implemented AC-system performed fair to good up to excellently on all 10 items. This suggests that AC is ready for a broader application in the field, bearing the potential for new opportunities in assessment. With regard to the accuracy increase by clustering as well as the poorest performances in terms of Kappa agreement (cf. Table 2), three items stuck out as being partially inadequate for the conducted AC: Item 3, *Interpret the author's intention*, Item 5, *Evaluate stylistic element*, and Item 1, *Explain protagonist's feeling*. While for the latter the reasons for AC failures were not obvious, responses on the first two of these items were partially coded improperly by AC because their correctness is crucially determined by other linguistic elements than just naming specific semantic concepts. In Item 3, *Interpret the author's intention*, the fact whether a response is correct or incorrect can be reliant on word order and relations, which is not detected by our approach to AC due to the use of the *bag of words*–paradigm, stemming, and stop word removal that often result in neglecting such information. This was also

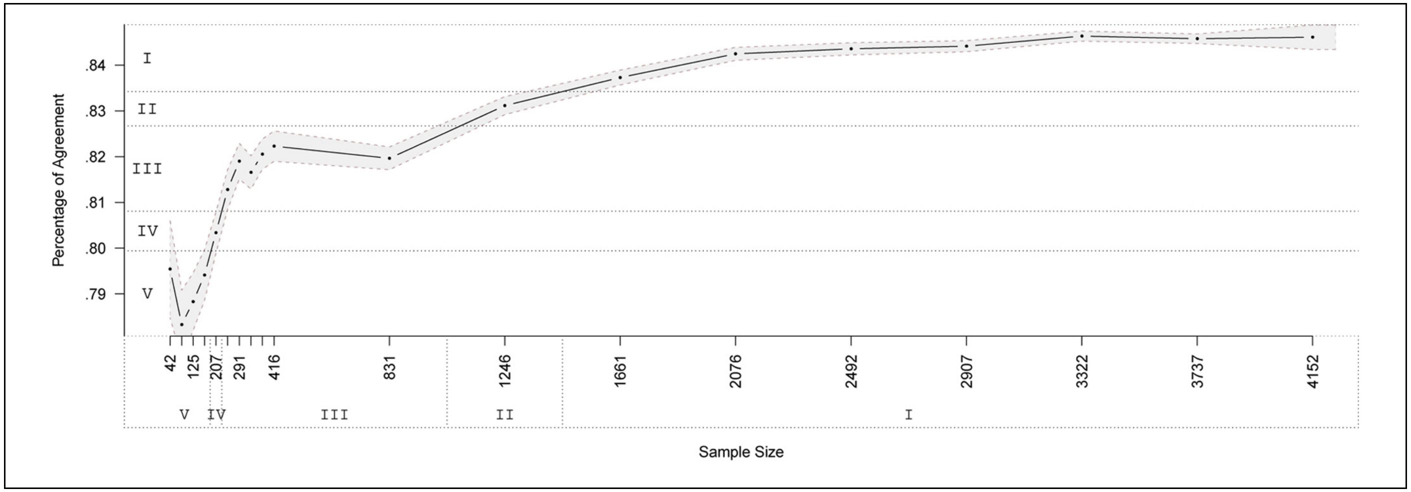

Figure 3. Results of the sample size simulation.
Note. The grey shaded area illustrates the means' 95% confidence intervals.

similarly true for Item 5, *Evaluate stylistic element*, in which responses' correctness is sometimes determined by synsemantic words (i.e., words that are not meaningful without other [autosemantic] words or a context). In this item, words such as "how,""what," and "why," as well as which sentence subject they refer to carry important information. In contrast to that, most of the analyzed items were coded properly by AC mainly taking the responses' semantics into account. Not only Item 4, *List recall*, was automatically coded excellently, being an extreme case of demanding isolated semantic concepts but also other items, such as Item 10, *Science*, and Item 6, *Verbal production*, which require a complex response for solving it, were coded satisfactorily by AC. Beside the reading items and the just mentioned item assessing scientific literacy this was also true for Item 9, *Math*. Hence, this kind of AC is not limited to reading as a construct heavily based on natural language but it is also adequate to more formal domains when using verbalizable contents.

Furthermore, we showed that some methods and parameter values can be used interchangeably (e.g., number of LSA dimensions) whereas others significantly affect performance (e.g., LSA outperformed ESA, which we believe appears because spaces built via LSA are optimized for a single item and in turn are experts of the item content, whereas an ESA space is based on a universal corpus such as the whole Wikipedia and, thus, carries a lot of irrelevant information for one specific item so that it is less sensitive for the relevant information). Importantly, we also showed that the optimal performance is not attained by a fixed set of methods and parameter values but by finding the most appropriate ones for the data. As a rule of thumb, those described in the section *Proposed Collection of Methods for Automatic Coding* were always within the best performing ones in our analyses: LSA for semantic space building (with at least 100 but mostly used 300 dimensions), cosine as distance metric and Ward (.D2) as agglomeration method. Spelling correction was shown not to be necessary for all of the 10 PISA items, which of course again vastly varies by the test takers' average spelling skills and words the item content evokes. Experience shows that only a thoroughly developed and adapted spelling correction component should be applied; otherwise performance might even decrease by wrong "corrections."

While improving with more data in general, AC can also be applied in studies with smaller data than large-scale assessment data. There was a nonlinear relationship with an exponential performance increase in the range of typical sample sizes of social science studies, at about $n = 250$, and an asymptotic development toward optimal performance starting at about $n = 1{,}700$, which is about the typical sample size of studies for psychological instrument development. The item we used for this simulation was a medium complex one in terms of evoked language diversity and, thus, should be representative. For items with less or more diversity, this curve will be compressed or stretched, respectively.

The proposed methods should be seen as a baseline still to be improved in the future. The unsupervised clustering approach is very conservative because during clustering, the responses' classes are not considered at all. Yet this is advantageous as completely unsupervised applications of AC is conceivable (e.g., if textual data are only meant to be categorized for analytical purposes; for more potential applications see the following list). In other contexts, so-called *Support Vector Machines* (e.g., see Theodoridis & Koutroumbas, 2009) constitute another machine learning method that will mostly outperform the proposed clustering approach. In the future, we intend to study the gain of semisupervised clustering in AC (e.g., see Witten et al., 2011).

The computational costs of the proposed AC are very low as soon as the model has been built once, and it should easily be possible to incorporate AC into existing procedures of large-scale assessments as well as into smaller studies. As the proposed methods appear to perform well, we want to encourage researchers as well as practitioners to use open-ended response format where it is appropriate and point out the following eight ways of using AC as examples that might improve existing and open doors to new assessment methods:

1. *Exploration*. Efficient exploration of textual data by automatic categorization is possible (e.g., for data sifting or to set up coding guidelines).
2. *Automatic Coding/Scoring*. By means of classified training data, codes can be assigned to each category (e.g., category of correct responses).
3. *More Differentiating Assessment*. Responses of different clusters may carry different information about the test taker. For instance, different clusters labeled all as correct may contain responses with different lines of reasoning. This might also lead to the possibility to synchronously assess a second construct enriching the principal construct (e.g., personality traits in a test which actually tests skills).
4. *Coding Guidelines Validity Check*. Multiple choice distractors are typically checked for their correlation to the construct. Now, for example, response clusters can be checked analogically for an unexpected high number of highly able test takers in a cluster that is interpreted as comprising incorrect responses. Such a case might reveal invalid coding guidelines.
5. *Control Loop for Human Coding*. If sticking to human coding, responses that are coded differently by human and computer can be revisited by a human (reducing errors from exhaustion, etc.).
6. *Reduced Human Coding*. Humans could only code, for example, 7 to 8 prototypical responses per cluster, which are defined as the most similar to the cluster centroid. In large-scale assessments, such as the presented PISA data, it then would not be necessary anymore to code more than 4,100 responses (per item!), but only, for instance, 30 clusters times 8 prototypical responses, resulting in 240 responses (an effort reduction of about 94%).
7. *Computer Adaptive Testing*. Adaptive testing requires to instantly code test taker responses. Therefore, usually closed response formats are used. Using AC expands Computer Adaptive Testing's scope to open-ended response format.

8. *Improving Assessment While Assessing*. Some test taker responses lack one single bit of information to exceed the threshold from incorrect to correct. But does this mean the test taker is not able to access this last bit? Since we are typically interested in the *latent* construct we indeed should distinguish between those test takers who *are* able to give us the missing bit and those who are *not*. It might turn out that a cluster contains such kind of responses; that means, there could be a cluster of responses that are almost correct but miss the last important bit of information. Automatic coding employed in computer-based assessment would then give the opportunity to probe the test taker for the missing bit. Of course, the implications of adding such a mechanism would need to be thoroughly studied. Nevertheless, it might be one more way to improve measures.

Limitations and Future Directions

We used German data only, but it should entail only a small to medium effort to apply the methods to further languages, since we utilized only baseline NLP methods that are available off-the-shelf for many different languages; as a showcase, being one implementation of one of many stemming algorithms, Snowball is available for six Germanic, six Italic, two Slavic, and two Uralic languages as well as for Turkish, Irish, Armenian, and Basque (cf. Porter, 2001). This is very interesting for international (large-scale) studies as using one coding system for multiple languages probably would strongly increase coding consistency between test languages, although the feasibility for all participating test languages would need to be examined.

Moreover, we only analyzed dichotomously coded items, whereas the used AC is also applicable to polytomous ones. For polytomously coded items it will be interesting to investigate the relationship between AC performance and sample size again with respect to the number of coding levels. For PISA 2012, only paper-based data were available that needed to be transcribed and, thus, is a possible source of noise due to typos. Also, computer-based assessed responses typically differ from paper-based assessed ones on the language level. We will soon be able to study the differences to computer-based data with the upcoming PISA cycle 2015. Furthermore, subgroup analyses like those of Bridgeman, Trapani, and Attali (2012) are needed to find whether this kind of AC performs equally for different test taker groups (native and nonnative speakers, high and low performers, different countries, etc.). Finally, it is worthwhile to study the possibility of using other external criteria for learning than human codings. Especially in the context of semisupervised learning it might be possible to use existing coding guidelines as learning criterion.

The newly implemented software used for analyses still lacks an interface that will be developed shortly in the context of an associated research group. As soon as ready, the software will be made freely and openly available and will enable researchers and practitioners to conduct AC on their data.[5] The software will not only contain the methods described here but also a more sophisticated NLP method called *Textual*

Entailment (cf. Androutsopoulos & Malakasiotis, 2010), which overcomes shortcomings of the *bag of words*–approach by performing inferences.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

Notes

1. Wikipedia dumps are available at http://dumps.wikimedia.org/
2. Building an analogy to factor analysis, the extracted semantic concepts would correspond to the factors in factor analysis, contexts (documents) would represent variables, and terms would stand for persons. The resulting matrix could be interpreted accordingly in that a single value of a term's vector would be the variable's factor loading, carrying the information of how semantically important the term (variable) reveals to be for the semantic concept (factor).
3. Cluster analyses comprise a large method family with several adaptations. Only one commonly used is described here.
4. As implemented in R's hclust-function, the analysis distinguishes Ward.D2, which squares cluster distances before updating, and Ward.D, which does not.
5. If interested, you can check the corresponding author's website (http://zib.education/en/pisa/mitarbeiter-pisa/fabian-zehner.html) for reference to the upcoming software's website.

References

Androutsopoulos, I., & Malakasiotis, P. (2010). A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research, 38*, 135-187. doi:10.1613/jair.2985

Bär, D., Zesch, T., & Gurevych, I. (2013). DKPro Similarity: An open source framework for text similarity. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (pp. 121-126). Sofia, Bulgaria: Association for Computational Linguistics.

Bejar, I. I. (2012). Rater cognition: Implications for validity. *Educational Measurement: Issues and Practice*, *31*(3), 2-9. doi:10.1111/j.1745-3992.2012.00238.x

Birenbaum, M., & Tatsuoka, K. K. (1987). Open-ended versus multiple-choice response formats: It does make a difference for diagnostic purposes. *Applied Psychological Measurement, 11*, 385-395. doi:10.1177/014662168701100404

Borra, S., & Di Ciaccio, A. (2010). Measuring the prediction error. A comparison of cross-validation, bootstrap and covariance penalty methods. *Computational Statistics & Data Analysis, 54*, 2976-2989. doi:10.1016/j.csda.2010.03.004

Bridgeman, B. (1991). A comparison of open-ended and multiple-choice question formats for the quantitative section of the Graduate Record Examinations General Test. *ETS Research Report Series*, 1991(*2*), 1-25. doi:10.1002/j.2333-8504.1991.tb01402.x

Bridgeman, B., Trapani, C., & Attali, Y. (2012). Comparison of human and machine scoring of essays: Differences by gender, ethnicity, and country. *Applied Measurement in Education*, *25*(1), 27-40. doi:10.1080/08957347.2012.635502

Burrows, S., Gurevych, I., & Stein, B. (2014). The eras and trends of automatic short answer grading. *International Journal of Artificial Intelligence in Education*, *25*(1), 60-117.

Carlson, S. S., & Ward, W. C. (1988). *A new look at formulating hypotheses items*. Princeton, NJ: Educational Testing Service. doi:10.1002/j.2330-8516.1988.tb00268.x

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science, 41*, 391-407. doi:10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1> 3.0.CO; 2-9

Dennett, D. C. (1991). *Consciousness explained* (1st ed.). Boston, MA: Little, Brown.

Dumais, S. T. (1991). Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments, & Computers, 23*, 229-236. doi:10.3758/BF03203370

Ferrucci, D., & Lally, A. (2004). UIMA: An architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering, 10*, 327-348. doi:10.1017/S1351324904003523

Fisher, R. A. (1915). Frequency distribution of the values of the correlation coefficient in samples of an indefinitely large population. *Biometrika, 10*, 507-521. doi:10.1093/biomet/10.4.507

Fisher, R. A. (1921). On the probable error of a coefficient of correlation deduced from a small sample. *Metron, 1*, 3-32.

Fleiss, J. L. (1981). The measurement of interrater agreement. In J. L. Fleiss, B. Levin, & M. C. Paik (Eds.), *Statistical methods for rates and proportions* (pp. 598-626). New York, NY: John Wiley.

Gabrilovich, E., & Markovitch, S. (2007). Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence* (pp. 1606-1611).

Gabrilovich, E., & Markovitch, S. (2009). Wikipedia-based semantic interpretation for natural language processing. *Journal of Artificial Intelligence Research, 34*, 443-498.

Gurevych, I., Mühlhäuser, M., Müller, C., Steimle, J., Weimer, M., & Zesch, T. (2007). Darmstadt Knowledge Processing Repository based on UIMA. In *Proceedings of the First Workshop on Unstructured Information Management Architecture at Biannual Conference of the Society for Computational Linguistics and Language Technology*.

Jurgens, D., & Stevens, K. (2010). The s-space package: An open source package for word space models. In *Proceedings of the ACL 2010 System Demonstrations* (pp. 30-35). Uppsala, Sweden: Association for Computational Linguistics.

Landauer, T. K. (2011). LSA as a theory of meaning. In T. K. Landauer, D. S. McNamara, S. Dennis, & W. Kintsch (Eds.), *Handbook of latent semantic analysis* (pp. 3-34). Mahwah, NJ: Erlbaum.

Landauer, T. K., McNamara, D. S., Dennis, S., & Kintsch, W. (Eds.). (2011). *Handbook of latent semantic analysis*. Mahwah, NJ: Erlbaum.

Martin, D. I., & Berry, M. W. (2011). Mathematical foundations behind latent semantic analysis. In T. K. Landauer, D. S. McNamara, S. Dennis, & W. Kintsch (Eds.), *Handbook of latent semantic analysis* (pp. 35-55). Mahwah, NJ: Erlbaum.

Millis, K., Magliano, J., Wiemer-Hastings, K., Todaro, S., & McNamara, D. S. (2011). Assessing and improving comprehension with latent semantic analysis. In T. K. Landauer, D. S. McNamara, S. Dennis, & W. Kintsch (Eds.), *Handbook of latent semantic analysis* (pp. 207-225). Mahwah, NJ: Erlbaum.

Organisation for Economic Cooperation and Development. (2013). *PISA 2012 assessment and analytical framework: Mathematics, reading, science, problem solving and financial literacy*. doi:10.1787/9789264190511-en.

Organisation for Economic Cooperation and Development. (2014). *PISA 2012 results: What students know and can do: Student performance in mathematics, reading and science* (Vol. *1*, Rev. ed.). doi:10.1787/9789264201118-en.

Porter, M. (2001). *Snowball: A language for stemming algorithms*. Retrieved from http://snowball.tartarus.org/texts/introduction.html

Prenzel, M., Sälzer, C., Klieme, E., & Köller, O. (2013). *PISA 2012: Fortschritte und Herausforderungen in Deutschland*. Münster, Germany: Waxmann.

R Core Team. (2014). *R: A language and environment for statistical computing*. Retrieved from http://www.R-project.org/

Rasch, D., Kubinger, K. D., & Yanagida, T. (2011). *Statistics in psychology using R and SPSS*. Chichester, England: John Wiley. doi:10.1002/9781119979630

Rauch, D. P., & Hartig, J. (2010). Multiple-choice versus open-ended response formats of reading test items: A two-dimensional IRT analysis. *Psychological Test and Assessment Modeling, 52*, 354-379.

Rupp, A. A., Ferne, T., & Choi, H. (2006). How assessing reading comprehension with multiple-choice questions shapes the construct: A cognitive processing perspective. *Language Testing, 23*, 441-474. doi:10.1191/0265532206lt337oa

Salton, G., & Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science, 41*, 288-297. doi:10.1002/(SICI)1097-4571(199006)41:4<288::AID-ASI8> 3.0.CO;2-H

Szarvas, G., Zesch, T., & Gurevych, I. (2011). Combining heterogeneous knowledge resources for improved distributional semantic models. *Computational Linguistics and Intelligent Text Processing, 6608*, 289-303. doi:10.1007/978-3-642-19400-9\textunderscore23

Theodoridis, S., & Koutroumbas, K. (2009). *Pattern recognition* (4th ed.). Burlington, MA: Academic Press. doi:10.1016/B978-1-59749-272-0.50003-7

Ward, J. H. J. (1963). Hierarchical grouping to optimize an objective function. *American Statistical Association Journal, 58*, 236-244. doi:10.1080/01621459.1963.10500845

Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data mining: Practical machine learning tools and techniques* (3rd ed.). Burlington, MA: Morgan Kaufmann. doi:10.1016/B978-0-12-374856-0.00023-7

Zesch, T., Müller, C., & Gurevych, I. (2008). Extracting lexical semantic knowledge from Wikipedia and Wiktionary. In *Proceedings of the 6th International Conference on Language Resources and Evaluation* (pp. 1646-1652).