Impact of Corpus Size and Dimensionality of LSA Spaces from Wikipedia Articles on AutoTutor Answer Evaluation

Zhiqiang Cai
The University of Memphis
365 Innovation Drive, Suite 410
Memphis, TN 38152-3115, USA
zcai@memphis.edu

Arthur C. Graesser
The University of Memphis
365 Innovation Drive, Suite 403
Memphis, TN 38152-3115, USA
graesser@memphis.edu

Leah C. Windsor
The University of Memphis
365 Innovation Drive, Suite 403
Memphis, TN 38152-3115, USA
lcwells@memphis.edu

Qinyu Cheng
The University of Memphis
365 Innovation Drive, Suite 410
Memphis, TN 38152-3115, USA
qcheng@memphis.edu

David W. Shaffer
University of Wisconsin-Madison
Aalborg University-Copenhagen
1025 West Johnson St
Madison, WI 53706-1706, USA
dws@education.wisc.edu

Xiangen Hu
University of Memphis
Central China Normal University
365 Innovation Drive, Suite 403
Memphis, TN 38152-3115, USA
xhu@memphis.edu

ABSTRACT

Latent Semantic Analysis (LSA) plays an important role in analyzing text data from education settings. LSA represents meaning of words and sets of words by vectors from a *k*-dimensional space generated from a selected corpus. While the impact of the value of *k* has been investigated by many researchers, the impact of the selection of documents and the size of the corpus has never been systematically investigated. This paper tackles this problem based on the performance of LSA in evaluating learners' answers to AutoTutor, a conversational intelligent tutoring system. We report the impact of document sources (Wikipedia vs TASA), selection algorithms (keyword based vs random), corpus size (from 2000 to 30000 documents) and number of dimensions (from 2 to 1000). Two AutoTutor tasks are used to evaluate the performance of different LSA spaces: a phrase level answer assessment (responses to focal prompt questions) and a sentence level answer assessment (responses to hints). We show that a sufficiently large (e.g., 20,000 to 30,000 documents) randomly selected Wikipedia corpus with high enough dimensions (about 300) could provide a reasonably good space. A specifically selected domain corpus could have significantly better performance with a relatively smaller corpus size (about 8000 documents) and much lower dimensionality (around 17). The widely used TASA corpus (37,651 documents scientifically sampled) performs equally well as a randomly selected large Wikipedia corpus (20,000 to 30,000) with a sufficiently high dimensionality (e.g., *k*>=300).

Keywords

AutoTutor, LSA, TASA, Wikipedia, corpus size, dimensionality

1. INTRODUCTION

1.1 Latent Semantic Analysis in Education Data Mining

Text mining is one of the most important tasks in education data mining [21]. Education text data could be textual learning content presented to learners, essays from learners, solutions to problems, answers to questions, conversations between collaborators, and so on. Researches have shown that analyzing such text data is crucial for improving education quality and reducing education cost. For example, Graesser et al. [9] reported that scaling texts to match the reading level and reading style of learners could facilitate the learning process. Foltz et al. [19] showed that automatic essay grading could greatly reduce teachers' workload. Wiemer-Hasting et al. [24] and Graesser et al. [10] showed that automatic answer evaluation makes it possible for intelligent environments to give immediate feedback to learners' text inputs. LSA (latent semantic analysis) plays an important role in all these text analysis tasks.

LSA is a method that extracts the meaning of words from a large body of texts (corpus) [15]. The mathematics behind LSA is surprisingly simple. The extraction process is just counting the number of occurrences of each word in each document, resulting in a word-document matrix, with rows representing words and columns representing documents. Thus, each row of the matrix is actually a vector representation of a word in a high dimensional space (the number of dimensions equals the number of documents). The raw occurrence counts are usually transformed by certain weighting method, such as TFIDF or Log-Entropy (see e.g., [14, 16]). After the transformation, a matrix entry has a higher value if the corresponding word is unevenly distributed in the corpus and frequent in the document corresponding to the column the word entry is in. A dimension reduction technique, namely, singular value decomposition, is applied to the weighted matrix to produce vector representations for words (as well as documents) with lower dimensionality. Weighted sum of word vectors is often used to form vector representations of phrases, sentences, paragraphs and documents. Different weight algorithms and their effects can be found in McNamara et al. [17]. More details on LSA vector space generation can be found in Landauer et al. [15].

With vector representations, the similarity of the meaning of two texts can be computed as the cosine between two vectors. This similarity measure has been widely used in many applications. For example, Coh-Metrix (cohmetrix.com) measures text cohesion by computing the average LSA cosine between sentence vectors and paragraph vectors [11]. AutoTutor (autotutor.org) evaluates learners' text inputs by computing the cosine between the input text vector and the ideal answer vector [4]. The Intelligent Essay Assessor [19] uses LSA cosine between vectors of target essay and pre-scored essays as one of the most important predictor in automatic essay scoring.

The number of dimensions of LSA vector spaces, usually denoted by *k*, has been investigated by many researchers. The most influential study is probably the one published by Landauer and Dumais in 1997 [14]. They generated an LSA space from 30,473 encyclopedia articles and then applied the vectors in a TOEFL (Test of English as a Foreign Language) word comparison task. They found that the value of *k* had large impact on the LSA performance and the best choice was about 300. This value, *k*=300, has been used as a magic number in many later applications. However, researchers also reported a large range of optimal values of *k* (from 6 to over 1000), depending on the corpus used for generating the LSA space and the specific task the LSA was applied to. A long list of studies can be found in Bradford (2008) [2].

In addition to dimensionality, the size and the content of the corpus used for LSA space generation also influences the performance of LSA. Researchers reported the use of different corpora, such as Touchstone Applied Science Association (TASA) corpus (http://lsa.colorado.edu/spaces.html), the Corpus of Contemporary American English (COCA) [7], Encyclopedia, and so on. The size of reported corpora varied from hundreds to hundreds of thousands of documents. Some studies reported the optimal values of *k* for different corpora with very different sizes. For example, Kontostathis (2007) [13] reported a study on 7 corpora with sizes varying from 1033 to 348,566 documents. While the optimal value of *k* for each corpus was reported, no corpus size effect was considered. A recent study reported by Crossley et al. (2017) [6] showed significant performance differences of two corpora with different sizes (44K vs 55K documents) and the same *k* =300, assuming 300 is the optimal value. However, they did not consider the real optimization of the value *k* for each corpus. Furthermore, the two corpora, TASA and COCA, contain different kinds of articles. It is not clear whether it was the corpus size or any other document features (e.g. genre, readability, concreteness, cohesion, formality, etc.) that caused the performance difference.

The size of data that needs to be stored for the word vectors of an LSA space is proportional to the product of number of words and the value of *k*. When performance is guaranteed, a small corpus with a small *k* would save both storage and computation cost in using LSA. Searching for an optimal corpus size to generate an LSA space is a task that needs to be accomplished together with the optimization of *k*. That is, it is an optimization problem involving two parameters, which unfortunately is more difficult and computationally more expensive. An added difficulty is to control other document features when the corpus size varies. One way to control the document features is to systematically sample articles from a particular single source to form corpora of different sizes. In this paper, we will focus on sampling corpora from Wikipedia.

1.2 Latent Semantic Analysis for AutoTutor Answer Evaluation

AutoTutor is a conversational intelligent tutoring system. Since the late 1990s, many AutoTutor systems have been developed, targeting different domains, including computer literacy, physics, critical thinking, and electronics. AutoTutor has become an ideal platform for collecting text data from learners and a testbed for natural language processing technologies.

AutoTutor helps learners learn by holding a conversation between computer agents (one or more) and human learners, targeting the solution to specific problems. AutoTutor usually starts a conversation with a deep "main question", to which the ideal answer is often about a paragraph in length. The goal of the conversation is to help a learner construct an answer semantically equivalent to the prepared ideal answer. To learners who cannot meet the goal at the first try, AutoTutor asks follow-up questions that target missing information. There are two types of such AutoTutor follow-up questions. One is called "hint", to which the answer is about a sentence long. Another is called "prompt", to which the answer is usually a word or a phrase. An AutoTutor conversation ends either when the goal is met or the prepared questions are exhausted. The following is an excerpt of an AutoTutor conversation in a Newtonian physics system.

AutoTutor [main question]: *Suppose a boy is in a free-falling elevator and he holds his keys motionless right in front of his face and then lets go. What will happen to the keys? Explain why.*
Student [first try]: *The keys will move up since the elevator is falling and the keys are in the elevator, the keys will go in the opposite direction of the free falling elevator.*
AutoTutor [feedback]: *Umm, that is an interesting answer!*
AutoTutor [hint]: *What can you say about the vertical velocity of the keys?*
Student [response to hint]: *The vertical velocity will be at a constant acceleration of 9.8.*
AutoTutor [feedback]: *OK. See if you can get this.*
AutoTutor [prompt]: *The boy and his keys have the same initial vertical what?*
Student [response to prompt]: *Velocity.*

In each conversation turn, AutoTutor evaluates the learner's input and makes decisions on the selection of feedback type and the next move (asking a new question or ending the conversation). With the help of regular expressions, LSA plays an important role in matching the meaning of the learner's input with prepared answers. The mathematical AutoTutor assessment models, normally a combination of LSA and regular expressions, can be found in Cai et al. (2011) [5].

When an AutoTutor application is developed, an LSA space generated from a domain specific corpus is often needed, because the meaning of words may differ from domain to domain. For example, the meaning of the word "force", according to Oxford Dictionary (https://en.oxforddictionaries.com) could be defined as "strength or energy as an attribute of physical action or movement" or "an organized body of military personnel or police". When an LSA space is generated, the meaning of a word can usually be observed by the "nearest neighbors", namely, the words with highest LSA cosine with the target word. Table 1 shows 5 nearest neighbors of the word "force" from three different LSA spaces: a Wikipedia space targeting Newtonian physics articles (4000 documents, 17 dimensions), a randomly sampled Wikipedia space (4000 articles, 17 dimensions) and the TASA corpus (37651 articles, 300 dimensions). It looks obvious that the meaning of "force" in the targeted corpus and TASA is more of the sense in Newtonian physics, while in the random Wikipedia space, the meaning is more of the sense in military.

Table 1. Nearest neighbors of "force" in different spaces

Corpus	Docs	Dim	Nearest Neighbors
Targeted	4000	17	exert, act, pull, experience, push
Random	4000	17	belligerent, offensive, gun, command, patrol
TASA	37651	300	unbalanced, exert, centripetal, turntable, Newton

It has long been believed that the performance of LSA depends on the selection of corpus. Cai et al [5] showed that, with a well selected corpus, LSA could be used together with regular expressions to build a model that evaluates learners' responses in AutoTutor almost as good as human. However, it has never been reported about the combined impact of the article selection, the size of the corpus and the optimization of *k* for the LSA component.

1.3 Wikipedia as Document Source for Corpus Sampling

In order to investigate the impact of document selection and corpus size, we need a reliable document source that contains enough many articles for different domains. Wikipedia (Wikipedia.org) is an ideal source for this. By the end of 2017, the English Wikipedia had about 5.6 million content articles, containing almost everything. New articles are still being added.

There are reports on LSA spaces generated from Wikipedia. For example, Ştefănescu et al. (2014) [22] compared the performance of Wikipedia spaces with TASA spaces on a word similarity task. However, they did not consider the "domain" specificity and the impact of the corpus size. They took all documents in Wikipedia as a whole for LSA space generation, taking into account of different filtering strategies, resulting in huge spaces.

1.4 Rational and Research Questions

Researchers have believed that the corpus used to generate an LSA space should align with the targeted domain. Gotoh et al. (1997) [8] showed a typical way of constructing a domain specific corpus: finding articles labeled in a category, such as "natural science", "world affairs", "arts", and so on. The targeted domain is then represented as a mixture of such categories. People are often convinced that domain specific spaces are needed from seeing "nearest neighbors" that show different meaning representations (see Table 1). However, several questions remain unanswered. For a given task, is it really necessary to generate a domain specific space? In other words, does a domain specific space perform significantly better than a generic space? A related question immediately emerges: how do we measure the "domain specificity"? How do we know the degree to which a corpus is targeting a given domain? Furthermore, what do we mean by a "domain"? How should a domain be defined or specified? There are also practical application questions related to this. For example, would a domain specific space save storage and computation costs with better or equivalent accuracy in performing a given task? Answers to these questions are important. If we know a generic space (e.g. TASA) can work as well as a domain specific space, we will not need to spend time and resources to generate new spaces. Domain spaces are needed only if they perform significantly better or can save storage and computation time without sacrificing performance.

2. METHOD

2.1 AutoTutor Data

We compared the performance of LSA spaces on evaluating learners' responses to a Newtonian physics AutoTutor. The data contained responses of college students to 10 problems about Newtonian physics. Table 2 shows the number of hints and prompts and the number of responses in each of the 10 problems. There were 114 hints and 133 prompts in total. This resulted in 4941 hint responses and 2643 prompt responses. On average, there were about 43 responses per hint and 20 responses per prompt. The reason why there were more hint responses was that AutoTutor conversations started with a hint, followed by a prompt, then another hint followed by another prompt, and so on. An AutoTutor conversation ended when a learner's responses covered all aspects of the ideal answer. Thus, if the conversation ended after a prompt, the number of hint responses and prompt responses in that conversation would be the same. However, if the conversation ended after a hint, the number of hint responses would be one more than the number of prompt responses in that conversation. The ratio of prompt responses to hint responses depends on the number of "hint-prompt" cycles that occurred in the conversation. The fact that there were more than twice of hint responses than prompt responses indicates that many conversations ended after the first or second hint question.

Table 2. Number of hint and prompt responses

Problem	Hints	Hint responses	Prompts	Prompt responses
Pumpkin	16	865	12	299
Sun and earth	6	186	1	23
Free key fall	12	969	10	403
Neck injury	11	308	12	272
Clown juggling	15	540	30	481
Car collision	11	431	6	86
Packet drop	13	801	9	285
Container mass	10	454	15	403
Clay balls	11	213	25	264
Car towing	9	174	13	127
Total	**114**	**4941**	**133**	**2643**

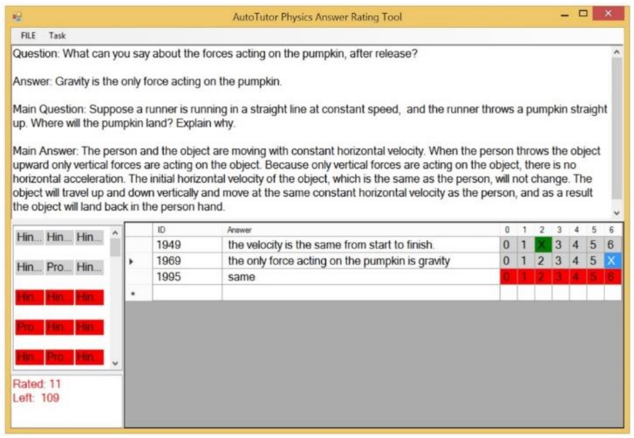
According to the design of AutoTutor, a hint question targets an answer about a sentence long and the answer to a prompt question is usually a word or a phrase. Table 3 shows that the hint responses in our data set were about 8 words on average; and the prompt responses were about 3 to 4 words on average. Penumatsa et al. (2006) [18] showed that the cosine values are length dependent. That is, longer texts tend to yield larger cosine values. Cai et al. (2016) [4] reported that LSA performed differently on hint responses and prompt responses. Their explanation was that the hint questions and prompt questions had different "uncertainties". The responses to a question with higher uncertainty would be more divergent and thus more difficult to assess. Following this, we investigated LSA performance on hint and prompt responses separately.

Table 3. Hint and prompt answer/response lengths

Answer	N	Mean	Std
Hint Ideal	112	10.64	3.59
Hint Responses	4861	8.01	8.82
Prompt Ideal	125	3.53	1.06
Prompt Responses	2603	1.64	3.06

2.2 Human Rating

The student responses were rated by two experts; one was a full professor and the other was a graduate student. Both raters had background in computer engineering and had good understanding of Newtonian physics. A rating tool was built to facilitate the rating process (see Figure 1). At the middle left panel of the tool, there is a list box for raters to choose hints and prompts. At the top panel, there is a text box that displays a hint/prompt question, together with its associated main question and their answers. The data table at the bottom right panel shows all student responses and 7 rating options. A response is scored by a click on an option. A "0" means the response is not an answer to the question at all, such as "what", "I don't know", "what do you mean", and so on. A "1" indicates a response that has no semantic similarity to the prepared ideal answer and "6" indicates a perfect answer. Red and gray colors are used to mark unrated and rated items, respectively. At the bottom left corner, there is a text box that shows the number of items already rated and the number of items that are to be rated. This tool helped the raters more easily and accurately rate the responses.



Question: What can you say about the forces acting on the pumpkin, after release?

Answer: Gravity is the only force acting on the pumpkin.

Main Question: Suppose a runner is running in a straight line at constant speed, and the runner throws a pumpkin straight up. Where will the pumpkin land? Explain why.

Main Answer: The person and the object are moving with constant horizontal velocity. When the person throws the object upward only vertical forces are acting on the object. Because only vertical forces are acting on the object, there is no horizontal acceleration. The initial horizontal velocity of the object, which is the same as the person, will not change. The object will travel up and down vertically and move at the same constant horizontal velocity as the person, and as a result the object will land back in the person hand.

ID	Answer	0	1	2	3	4	5	6
1949	the velocity is the same from start to finish	0	1	X	3	4	5	6
1969	the only force acting on the pumpkin is gravity	0	1	2	3	4	5	X
1995	same	0	1	2	3	4	5	6

Rated: 11
Left: 109

Figure 1. Rating tool.

From the 7584 responses, 120 were randomly sampled as training corpus. After rating the training corpus, two raters discussed the rating criteria and independently rated the rest of the items. Table 4. shows the correlations between the two raters. The correlations were about 0.82, which indicate that there were some disagreements between the two raters. In other words, even for human experts, such evaluation tasks are sometimes difficult. We had thought that answers to prompts should be easier to evaluate than answers to hint. However, the correlation of two raters' ratings on prompts is only slightly higher. The Fisher transform [12] showed that the Z value of the two correlations (hints vs prompts) was 0.90 (*p*=0.369), which indicates that the difference is not significant. It should be noted that this Z value is for two independent correlations from different samples. There is another Z-test for dependent correlations, which will be used in the later part of this paper. The Z transform showed that the human rates agreed on hint and prompt responses similarly. That indicates that human raters did not experience more difficulty in evaluating hint responses than prompt responses.

Table 4. Correlations between ratings of two raters.

Question Type	N	Correlation
Hint	4861	0.820
Prompt	2603	0.827
All	7464	0.828

2.3 Sampling Corpora from Wikipedia

2.3.1 *Seeding method for sampling domain specific corpus*

Our goal was to investigate whether or not a "domain specific" corpus generates an LSA space with higher performance for our tasks. However, it is hard to quantify what a "domain specific" corpus really is. Many researchers used corpora that showed obvious domain labels. For example, MED corpus is for "Medical", CISI corpus for "Information Science" [3], COCA for "Contemporary American English" [7], and so on. However, we don't really know how specific these corpora are with respect to the labeled domain.

Our way of handling this problem starts from specifying a domain by a seed corpus – a small number of documents representing the targeted domain. The seed corpus could be the sections of a book, a small collection of articles focusing on a specific topic, or just some documents that are under analysis.

Once a seed corpus is identified, we extract keywords out of the seed corpus and assign a "keyness" value to each keyword. Thus, a "domain" is represented by the keyness assignment to the domain vocabulary. This is similar to the idea in topic modeling, where a topic is represented by probability distribution on a word list (see, for example, [1]).

The word keyness computation is then applied to compute document keyness by averaging the keyness of the words in a document. To search documents from a large document source (such as Wikipedia), we compute the keyness of each document. The documents in the source are then ranked by the document keyness. We select the high ranking documents from the source as a domain corpus. We call this process the "seeding method."

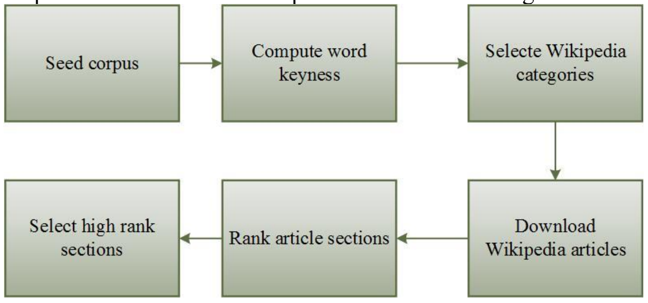

Seed corpus
Compute word keyness
Selecte Wikipedia categories
Download Wikipedia articles
Rank article sections
Select high rank sections

Figure 2. Illustration of seeding method.

Figure 2 visually illustrates the process of selecting documents from Wikipedia articles. It is difficult to directly evaluate the validity of this process. A possible way is to present a sample of documents to experts and see what proportion of documents are highly relevant to the desired domain. We do not do so in this paper. Instead, we evaluate this process by comparing the LSA performance of selected corpora with randomly sampled corpora. Our logic is simple: if domain specificity matters and the selected corpora work better than random corpora, then the seeding method is valid.

2.3.2 Computing word keyness

To compute the keyness of words, we considered two factors. First, a high keyness word should not be very common in general use. To quantify this, we used the log-entropy weight from a general reference corpus, TASA, as a measure of how common a word is in general use:

$$E(w) = 1 + \frac{\sum_{i=1}^{N} p_i(w) log p_i(w)}{log N}$$

where

$$p_i(w) = \frac{frequency\ of\ w\ in\ document\ i}{total\ frequency\ of\ w\ in\ the\ corpus}$$

In the above equations, N is the number of documents in TASA, which is 37,651. The log-entropy weight, $E(w)$, ranges from 0 to 1. A value close to 0 indicates that the word w is evenly distributed in TASA corpus, such as function words. A value close to 1 indicates that the word distributed unevenly in the corpus. More detailed information about entropy use can be found in LSA publications (for example, Martin et al. (1994) [16]).

Another factor we considered was that a high keyness word should be highly frequent in the seed corpus. We used the normalized logarithm of frequency to quantify this. The final word keyness with respect to the seed corpus was computed as the product of two values. One was the logarithm of the number of seed documents the word is in, divided by the logarithm of the total number of documents; and the other was the log-entropy weight of the word in TASA:

$$keyness = \frac{\log f(w)}{\log D} E(w)$$

where

$f(w)$: the number of seed documents the word w is in;

D: the total number of seed documents; and

$E(w)$: the log-entropy weight of the word w from TASA corpus.

2.3.3 Sampling a Newtonian physics corpus from Wikipedia

For this study, we used the 114 hint questions and the 133 prompt questions as seed corpus to compute the word keyness. This is such a small corpus that it only covered a small part of Newtonian physics. However, it provided a good starting point for us to find related categories from Wikipeida. Using the keyness equation, each word in the seed corpus was assigned a keyness. We ignored the words with keyness less than 0.01 and obtained 262 keywords. The top 10 keywords, together with their keyness values are listed below:

1) free-fall: 0.588
2) packet: 0.537
3) pumpkin: 0.526
4) acceleration: 0.496
5) velocity: 0.478
6) clown: 0.467
7) velocities: 0.449
8) horizontal: 0.427
9) keys: 0.424
10) headrest: 0.407

From the list above, we see that some words, such as “acceleration”, “velocity” are concepts of Newtonian physics. However, other words are specific to the 10 problems. To construct a corpus that has a wide coverage of Newtonian physics, we queried Wikipedia categories with these 262 keywords and obtained 154 associated categories. From these 154 categories we manually selected 16 categories that are highly related to Newtonian physics, as shown in the list below:

1) Acceleration
2) Change
3) Classical mechanics
4) Concepts in physics
5) Dynamics(mechanics)
6) Force
7) Gravitation
8) Kinematics
9) Mass
10) Mechanics
11) Motion
12) Physics
13) Systems
14) Temporal rates
15) Time
16) Velocity

In Wikipedia, each category is associated with a set of articles and a set of subcategories. For example, at the time this paper was written, the category “force” contained 67 articles (such as “force”, “friction”, “weight”, etc.) and 8 subcategories (such as “motion”, “fictitious forces”, “friction”, etc.) The above 16 selected categories served as “seed categories” of our query. We downloaded all articles from these 16 categories. Then we downloaded the articles from subcategories. Since each subcategory also contained subcategories, we could actually find a very large number of articles by following the subcategories of subcategories. In this study, we downloaded 30,000 articles.

We did not treat each article as a document of our corpus for LSA space generation. Instead, we used selected sections in the articles. Each Wikipeida article contained a definition section and many other sections. For example, the article “force” in physics contained 17 sections, such as “Development of the concept”, “Pre-Newtonian concepts”, “Newtonian mechanics”, etc. We computed the keyness of each section of each article by averaging the word keyness computed from the seed corpus. Words that did not appear in the seed corpus or with keyness less than 0.01 were ignored. Notice that, although the problem specific keywords, such as “packet”, “pumpkin”, “clown”, etc., had high keyness, since they are unlikely to appear in the articles from the selected categories, their effect in the section selection process was limited. The list below shows the number of sections of the top 10 keywords appeared in a corpus with 32,000 selected sections:

1) free-fall:73
2) packet:106
3) pumpkin:7
4) acceleration:1906
5) velocity:4424
6) clown:4
7) velocities:866
8) horizontal:1204
9) keys:124
10) headrest:1

Obviously, the Newtonian physics concepts, such as “velocity”, “acceleration”, etc., dominated the selection process. The problem specific terms, such as “headrest”, “clown” and “pumpkin” rarely appeared in the selected sections.

To avoid section length effect, we ignored any section with length < 50 words. For sections with words between 50 and 300, the section keyness was the average word keyness. For long sections with more than 300 words, the keyness was computed as the average over the first 300 words.

To compare the impact of the corpus size, we selected 5 corpora with size (number of sections) 2,000, 4,000, 8,000, 16,000 and 30,000. We name them NP2000, NP4000, …, NP30000, where "NP" stands for "Newtonian Physics". Each Newtonian physics corpus contained the highest keyness sections in the selected articles. Therefore, they were nested, namely, the sections of a smaller Newtonian physics corpus were all included in a larger Newtonian physics corpus. For example, NP8000 contained all sections of NP2000 and NP4000 (see Figure 3, left).

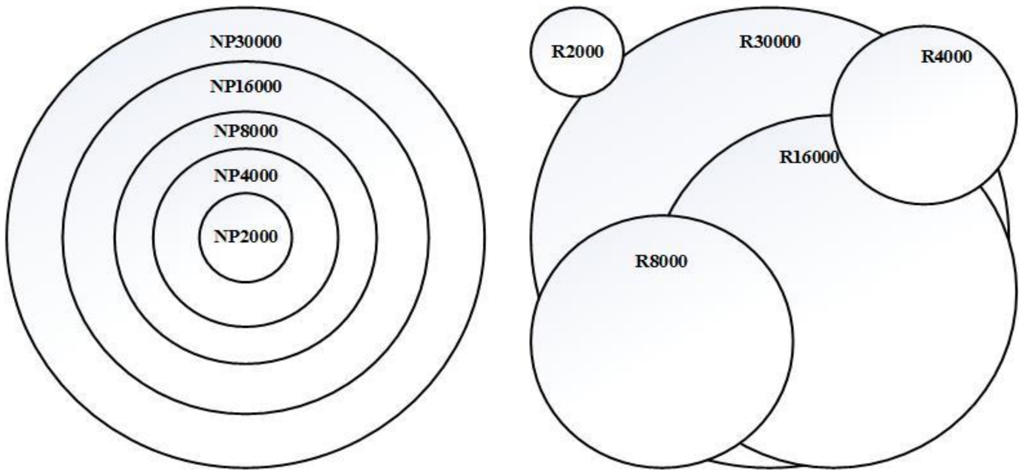


Figure 3. Illustration of nested domain corpora (left) and overlapped random corpora (right).

2.3.4 *Sampling a random corpus*

In order to compare the effect of the keyness-based sampling, we randomly sampled 5 corpora with same sizes as Newtonian physics corpora. The sampling process was similar to the Newtonian physics corpora sampling. The difference was that the seed keywords were 1000 words randomly sampled from TASA vocabulary. We downloaded 30,000 articles from the categories and their subcategories associated with the 1000 seed keywords. Then we randomly sampled sections from the 30,000 articles. Like NP sampling, sections with less than 50 words were ignored. The five random corpora were named, based on their sizes, as R2000, R4000, R8000, R16000 and R30000. The random corpora could be overlapped but not necessarily nested (see Figure 3, right).

2.4 LSA Spaces

A total of 11 LSA spaces were generated, 5 Newtonian physics spaces, 5 random spaces and a TASA space. The log-entropy weighting was applied to the word-document matrices. Function words and words appeared less than 3 documents in a corpus were ignored. The dimensions were all 1,000. In the rest of the paper, we will only refer to these 11 spaces. However, the similarities in each space were computed with varied dimensions. Mathematically speaking, different dimensions means different spaces. For example, NP8000 with 100 dimensions is a different space than NP8000 with 300 dimensions. However, in this paper, we refer to them as the "same space" and treat the dimension as a parameter in computing LSA similarities.

2.5 Evaluating AutoTutor Responses by LSA

For each of the above 11 spaces, the LSA semantic similarities between ideal answers and learners' responses were computed for the varying number of dimensions (k=2, 3, …, 1000). The performance of each space with each value of k was measured by the correlation between the LSA similarity and the average human rating on the responses. Cai et al [4] showed that LSA performances on hint questions and prompt questions are very different. Therefore, we considered the LSA performance on hint questions and prompt questions separately. Table 5 shows some example responses of a hint question, their LSA similarity to the ideal answer, and the human rating.

Table 5. Example of learners' responses to the hint question "How does the net force affect the car?". The ideal answer is "The net force exerted on the car results in an acceleration of two meters per second squared." LSA similarities were computed using TASA space, 300 dimensions. Human ratings are average scores of two raters.

Response	LSA	Human
Horizontally	0.16	1
it stays the same	0.16	1
it does not effect the car	0.19	1
it causes it to accelerate	0.49	2
the net force doesn't change and therefore when the mass is doubled the acceleration must be halved	0.51	5.5
The net force on the car is what causes it to accelerate	0.69	5.5
It causes an acceleration of two meters per second	0.71	5.5

The significance of performance differences were measured by Steiger Z-test [23], which is a statistic method for testing the significance of differences between two dependent correlations that share a variable in common. This is different from the independent correlation comparison that we used earlier. The Steiger Z-test compares correlation coefficients involving three variables. Assuming the three variables are A, B and C, with C as the shared common variable, the two correlations under comparison are:

- The correlation coefficient between A and C and
- The correlation coefficient between B and C.

To compare the two correlation coefficients, the correlation coefficient between A and B is also included in the computation, together with the number of data points, *N*. When the absolute value of Z is greater than 1.96, the two correlation coefficients under comparison are considered significantly different. In our study, C is the human rating whereas and A and B are two LSA similarities.

3. RESULTS

3.1 Impact of corpus size and number of dimensions for Newtonian physics spaces

Consider first the hint responses. Although the corpus sizes were very different among the five Newtonian physics spaces, the performance curves as functions of dimensions were surprising similar. They all had lowest performance at *k*=2, with correlations about 0.28. When the number of dimensions increased, the performance curves of all spaces quickly increased. The peak of about 0.425 was reached around *k*=17. The performance curves then dropped and reached a trough around *k*=128. After that, they grew up again and converged from about *k*=300 to a value about 0.40. Figure 4 shows the performance as functions of *k* for the NP spaces on hint responses from *k*=2 to *k*=1000. We used a logarithm scale on dimensions, following the method that Landauer et al. (1998) used when plotting the dimensionality effects on TOEFL tests.

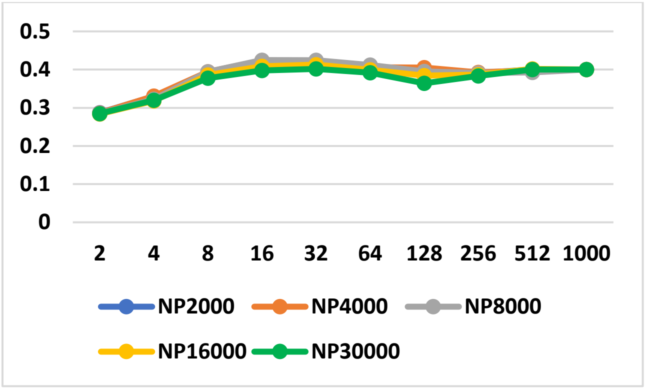


Figure 4. Performance of NP spaces on hint responses.

Although not very large, significant differences were observed among different spaces. The best performance on hint responses was NP8000 at *k*=17, with the highest correlation being 0.428. Z-test showed that, with the same *k* value, NP8000 performed significantly better than other spaces (see Table 6). The differences of correlations were from 0.01 to 0.036. This value, 0.428, was also significantly better than the performance of the same space NP8000 with *k* value less than 16 or greater than 64 (see Table 7).

Table 6. Z-test comparing performance of target spaces with fixed *k*=17 and varied corpus size on hints to the optimal target space (corpus size=8000, *k*=17) and performance (0.428). R-opt is the correlation with the optimal space. N=4861.

Space	Performance	R-opt	Z	p(2-tail)
NP2000	0.415	0.916	2.449	0.014
NP4000	0.418	0.964	2.837	0.004
NP16000	0.409	0.970	5.654	0.000
NP30000	0.392	0.934	7.607	0.000

Table 7. Z-test comparing performances of target spaces with fixed corpus size=8000 and varied *k* on hints to the optimal target space (corpus size=8000, *k*=17) and performance (0.428). R-8000 is the correlation to the optimal space. N=4861.

Dim	Performance	R-opt	Z	p(2-tail)
2	0.286	0.582	11.837	0
4	0.322	0.731	11.031	0
8	0.394	0.913	6.265	0
16	0.425	0.997	2.986	0.003
32	0.425	0.953	0.756	0.45
64	0.412	0.88	2.525	0.012
128	0.395	0.794	3.976	0
256	0.391	0.732	3.916	0
512	0.393	0.689	3.448	0
1000	0.396	0.655	3.001	0.003

The Newtonian physics spaces performed differently on prompt questions. The overall performance on prompts were higher than on hints. Also, the corpus size had a larger impact. NP8000 performed best overall. Two smaller spaces, NP2000 and NP4000, performed significantly worse. Larger spaces performed almost equally as well as NP8000. For *k*=2, the performance of smaller spaces was around 0.2, while for larger spaces, the performance was over 0.3. The performance curves for all spaces increased when the value of *k* increased. However, there was no early peak. At about *k*=24, the performance curves started to converge. The best performance for *k*=24 was again the space NP8000, which was 0.542. When the value of *k* further increased, the performance curves continuously and slowly increased. The maximum performance was at about *k*=300, which is 0.566 for larger spaces. The performance curves slowly dropped after *k*=300. At *k*=1000, the performance of small spaces was about 0.51 and the larger spaces around 0.54. Figure 5 shows the performance curves of the spaces as functions of *k* on prompt responses.

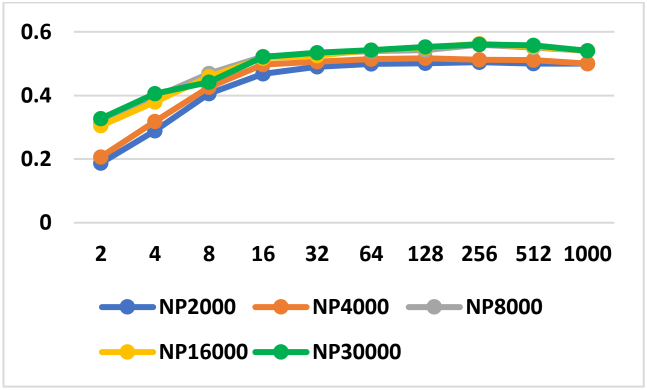


Figure 5. Performance of NP spaces on prompt responses

3.2 Comparing with Random Wikipedia Spaces and TASA Space

For small values of *k* (<32), Newtonian physics spaces performed much better (about 0.1 higher) than random spaces and TASA space on both hint responses and prompt responses. TASA space was worse than Newtonian physics spaces but better than random spaces. However, the performance curves of all large spaces converged to almost the same after about 300 dimensions. Figure 6 shows the performance of NP8000, R8000 and TASA on hint responses. Unlike Newtonian physics spaces, random spaces and TASA space did not have a peak performance. Instead, their performance curves continuously and slowly grew and converged.

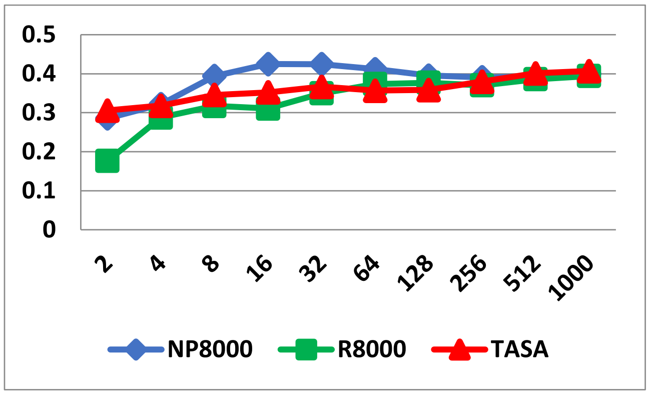


Figure 6. Comparing NP8000, R8000 and TASA on hint responses.

Figure 7 shows the performance of NP8000, R8000 and TASA spaces on prompt questions. Newtonian physics space NP8000 performed best, especially at around *k*=17. For lower dimensionality, TASA space was slightly better than random space. However, after *k*=32, the random space became slightly better than TASA.

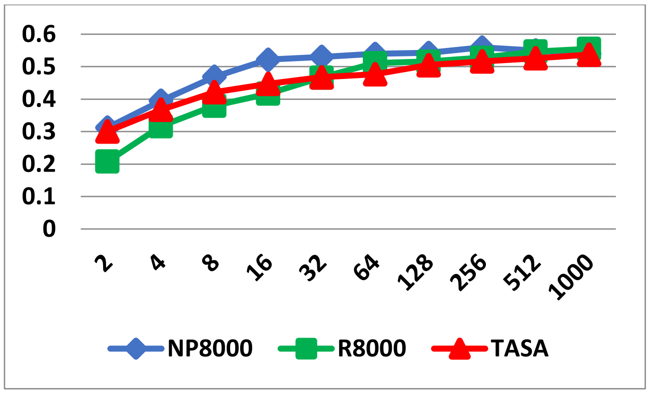


Figure 7. Comparing NP8000, R8000 and TASA space on prompt responses.

4. DISCUSSION AND FUTURE WORK

In AutoTutor applications, LSA similarities has been used as an important feature for building models to evaluate learners' responses. Every time a new application was created, a new "domain specific" spaces was generated. The so called "domain specific" spaces were usually generated from a corpus provided by domain experts. It was often unclear whether or not the documents in the domain corpus were sufficiently representative. That motivated us to explore the impact of document selection and corpus size, taking into account the optimal space dimensionality.

Instead of relying on experts' selections, we used Wikipedia as a universal source to select corpus for any domain. In this study, we used a method called "seeding method" to select Wikipedia articles based on a small seed corpus. Although the seeding method started with an automatic keyness computation and ended with automatic document ranking and selection, the method was not fully automatic, because, in the middle of the process, a manual Wikipedia category selection was involved. Because of this manual selection, the document ranking was constrained by the category selection. That is, the document ranking was computed only over a subset of Wikipedia articles. Although this reduced the searching cost, it is not clear how much better a space could be if the documents were selected from all Wikipedia articles. A fully automatic and inexpensive Wikipedia article selection algorithm apparently is still needed.

The seeding method was not directly evaluated. However, its effect has been shown by the fact that the selected spaces perform significantly better than random spaces. Yet, the seeding method might have room for improvement. Better keyness assignment and document ranking algorithms are possible. For example, the entropy based keyword extraction algorithm provided by Yang et al. [25] is a good candidate for more sophisticated keyness assignment algorithms. Even further, instead of keyness based document ranking, other methods without keyness assignment are possible. For example, a seed LSA space could be generated from the seed corpus. Then a small number of Wikipedia articles could be selected to form a slightly larger corpus. Then a larger LSA space is generated and more Wikipedia articles are added. Such an iterative process could be more expensive but may provide better LSA spaces.

This study revealed several interesting results about the impact of dimensionality. When we examined the performance, we did not expect that *k*=2 could provide a significant correlation. It turned out that 2 dimensional spaces (e.g. NP8000, TASA) could actually perform quite well (around 0.3). This fact is important because two dimensional vectors are easy for visualization. Therefore, if a 2-dimensional space could provide acceptable performance, it may be considered if visualization is a concern.

Another interesting finding is that the optimal *k* could be very small (e.g., 17 for NP8000). A small *k* implies low cost in both storage and computation. However, it may not be possible to identify the optimal *k* without dependent data, such as human ratings. When such data is not available, we certainly want to know what *k* is safe for use. This study showed that *k*=300 is a safe dimensionality for both hint and prompt response evaluation.

It seems obvious that there must be an optimal corpus size, which is not too small and not too large. If a corpus is too small, it may have two problems: 1) it cannot represent the desired domain and 2) it cannot provide enough semantic associations for generating meaningful vectors. If a corpus is too large, it will lose focus. This study shows that NP8000 is better than smaller and larger corpora. The problem, however, is that this optimal size is identified using human rated data. When human ratings are not available, a relatively large corpus would be safer.

TASA space has been widely used in LSA research, as discussed earlier. However, it has been an open question whether a domain specific LSA space would have better performance than a broader TASA space. This study shows that the performance of TASA space on AutoTutor tasks is close to random Wikipedia spaces of large enough corpora and high dimensionality. Even compared with well selected corpora, a TASA space with high dimensionality (e.g., *k*>=300) performs reasonably well on AutoTutor tasks. Therefore, for an application in English language, it should be safe to use TASA space (with *k*=300) when LSA is used for semantic comparison.

However, there are two problems in using TASA. The first problem is that there could be important domain specific terms that are not included in TASA corpus. Another problem is that TASA is an

English corpus. When a space for another language, such as Chinese, French, etc., is needed, there is no simple way to compose a TASA corpus in other languages. Sampling documents from Wikipedia using the seeding method is a good solution to these problems.

There are hundreds of Wikipedias in different languages. Sampling a Wikipedia corpus in any language is easy and free. The seeding method guarantees that the selected articles would include the keywords in the targeted domain. The seeding method also provides significantly better performance with relatively smaller spaces, a smaller vocabulary and a smaller number of dimensions. This means the that the seeding method helps reducing the cost of storage and computing while maintaining performance levels.

In our study, the performance of domain specific spaces could be approximated reached in random Wikipedia spaces or TASA space. The difference is that, domain specific spaces could perform well with very low dimensionality, while non-domain specific spaces need much higher dimensionality to get to the same level of performance. Therefore, the value of using domain specific spaces could be the possible use of low dimensionality. This may have important implications is other applications. For example, in deep learning on natural language processing, reliable low dimensional word embedding will save training cost and make trained models more generalizable.

To conclude, using seeding method and Wikipedia in LSA space generation has the following advantages:

- It guarantees domain keyword inclusion;
- The same method can be applied to all languages;
- It reduces cost of storage and computing; and
- It improves semantic evaluation accuracy.

Once again, LSA similarity is only one of the factors considered in evaluating AutoTutor responses. The correlation values, about 0.43 on hint responses and 0.56 on prompt responses, are still far away from human's agreement (*r*>=0.82). In order to further improve AutoTutor assessment accuracy, other evaluation methods are needed, such as regular expressions. Cai et al. (2016) [4] proposed an alternative way in computing the LSA similarity. Instead of comparing the responses with the author-prepared ideal answer, they compared it with group responses. As we mentioned earlier, combining regular expression with LSA would make a better assessment model. In other words, LSA similarity may be used as a very powerful predictor to build a model to simulate human rating. However, using LSA alone is usually not enough. LSA vectors could also be used as word embedding to train deep learning models [20]. We did not include such algorithms in this paper, because our focus is on the quality of spaces, not the quality of AutoTuotor assessment model.

5. ACKNOWLEDGMENTS

The research on was supported by the National Science Foundation (DRK-12-0918409, DRK-12 1418288), the Institute of Education Sciences (R305C120001), Army Research Lab (W911INF-12-2-0030), and the Office of Naval Research (N00014-12-C-0643; N00014-16-C-3027). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF, IES, or DoD. The Tutoring Research Group (TRG) is an interdisciplinary research team comprised of researchers from psychology, computer science, and other departments at University of Memphis (visit http://www.autotutor.org).

6. REFERENCES

[1] Blei, D.M., Ng, A.Y. and Jordan, M.I. 2015. Latent Dirichlet Allocation. *Journal of Machine Learning Research*. 3, (2015), 993–1022.

[2] Bradford, R.B. 2008. An empirical study of required dimensionality for large-scale latent semantic indexing applications. *Proceeding of the 17th ACM conference on Information and knowledge mining - CIKM '08*. (2008), 153.

[3] Buckley, C. 1985. Implementation of the smart information retrieval system. Ithaca.

[4] Cai, Z., Gong, Y., Qiu, Q., Hu, X. and Graesser, A. 2016. Making autotutor agents smarter: Autotutor answer clustering and iterative script authoring. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 10011 LNAI, (2016), 438–441.

[5] Cai, Z., Graesser, A., Forsyth, C., Burkett, C., Millis, K., Wallace, P., Halpern, D. and Butler, H. 2011. Trialog in {ARIES}: User input assessment in an intelligent tutoring system. *Proceedings of the 3rd IEEE International Conference on Intelligent Computing and Intelligent Systems*. March 2016 (2011), 429–433.

[6] Crossley, S.A., Dascalu, M. and Mcnamara, D.S. 2017. How important is size? An Investigation of Corpus Size and Meaning in both Latent Semantic Analysis and Latent Dirichlet Allocation. *FLAIRS Conference* (2017), 293–296.

[7] Davies, M. 2010. The Corpus of Contemporary American English as the first reliable monitor corpus of English. *Literary and Linguistic Computing*. 25, 4 (2010), 447–464.

[8] Gotoh, Y. and Renals, S. 1997. Document space models using latent semantic analysis. 7, (1997), 6–9.

[9] Graesser, A.C., Feng, S. and Cai, Z. 2017. Two Technologies to Help Adults with Reading Difficulties Improve their Comprehension. *Developmental perspectives in written language and literacy. In honor of Ludo Verhoeven*. E. Segers and P. van den Broek, eds. John Benjamin Publishing Company. 296–313.

[10] Graesser, A.C., Forsyth, C.M. and Foltz, P. 2017. Assessing conversation quality, reasoning, and problem-solving performance with computer agents. *The nature of problem solving: Using research to inspire 21st century learning*. 245–261.

[11] Graesser, A.C., McNamara, D.S., Louwerse, M.M. and Cai, Z. 2004. Coh-Metrix: Analysis of text on cohesion and language. *Behavior Research Methods, Instruments, & Computers*. 36, 2 (2004), 193–202.

[12] Hotelling, H. 1953. New Light on the Correlation Coefficient and its Transforms. *Journal of the Royal Statistical Society. Series B (Methodological)*. 15, 2 (1953), 193–232.

[13] Kontostathis, A. 2007. Essential Dimensions of Latent Semantic Indexing (LSI). (2007), 1–8.

[14] Landauer, T.K. and Dumais, S.T. 1997. A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. *Psychological Review*. 104, 2 (1997), 211–240.

[15] Landauer, T.K., Folt, P.W. and Laham, D. 1998. An introduction to latent semantic analysis. *Discourse processes*. 25, 2 (1998), 259–284.

[16] Martin, D.I., Bear, S. and Consulting, T. 1994. Mathematical Foundations Behind Latent Semantic Analysis. 35–55.

[17] McNamara, D., Cai, Z. and Louwerse, M. 2007. Optimizing LSA Measures of Cohesion. *Handbook of latent semantic analysis*. T.K. Landauer, D.S. McNamara, S. Dennis, and W. Kintsch, eds. Erlbaum. 379–400.

[18] Penumatsa, P., Ventura, M., Graesser, A.C., Louwerse, M., Hu, X., Cai, Z. and Franceschetti, D.R. 2006. The Right Threshold Value: What is the Right Threshold of Cosine Measure When Using Latent Semantic Analysis for Evaluating Student Answers? *International Journal on Artificial Intelligence Tools*. 15, 05 (2006), 767–777.

[19] Peter W. Foltz Lynn A. Streeter, K.E.L.T.K.L. Implementation and Applications of the Intelligent Essay Assessor.

[20] Riordan, B., Horbach, A., Cahill, A., Zesch, T. and Lee, C.M. 2017. Investigating neural architectures for short answer scoring. *$Bea17*. (2017), 159–168.

[21] Slater, S., Joksimović, S., Kovanovic, V., Baker, R.S. and Gasevic, D. 2017. Tools for Educational Data Mining: A Review. *Journal of Educational and Behavioral Statistics*. 42, 1 (2017), 85–106.

[22] Ștefănescu, D., Banjade, R. and Rus, V. 2014. Latent Semantic Analysis Models on Wikipedia and TASA. *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC'14)*. (2014), 1417–1422.

[23] Steiger, J.H. 1980. Tests for comparing elements of a correlation matrix. *Psychological Bulletin*.

[24] Wiemer-Hastings, P., Graesser, A.C., Harter, D. and Grp, T.R. 1998. The foundations and architecture of autotutor. *Intelligent Tutoring Systems*. 1452, (1998), 334–343.

[25] Yang, Z., Lei, J., Fan, K. and Lai, Y. 2013. Keyword extraction by entropy difference between the intrinsic and extrinsic mode. *Physica A*. 392, 19 (2013), 4523–4531.