Journal of Education and e-Learning Research
Vol. 10, No. 2, 111-118, 2023
ISSN(E) 2410-9991 / ISSN(P) 2518-0169
DOI: 10.20448/jeelr.v10i2.4449

Deep learning for the detection of acquired and non-acquired skills in students' algorithmic assessments

Floran Carvalho[1]
Julien Henriet[2]✉
Francoise Greffier[3]
Marie-Laure Betbeder[4]
Dana Leon-Henri[5]


(✉ Corresponding Author)

[1,2,4]*FEMTO-ST DISC Institute, Univ. Bourgogne-Franche-Comte, 16 route de Gray, F-25030 Besançon Cedex, France.*
[1]*Email: floran.carvalho@univ-fcomte.fr*
[2]*Email: julien.henriet@univ-fcomte.fr*
[4]*Email: marie-laure.betbeder@univ-fcomte.fr*
[3,5]*ELLIADD (EA 4661), Univ. Bourgogne-Franche-Comte, 30-32 rue Megevand CS 81807, F-25030 Besançon Cedex, France.*
[3]*Email: francoise.greffier@univ-fcomte.fr*
[5]*Email: dana.leon-henri@univ-fcomte.fr*

Abstract

This research is part of the Artificial Intelligence Virtual Trainer (AI-VT) project which aims to create a system that can identify the user's skills from a text by means of machine learning. AI-VT is a case-based reasoning learning support system can generate customized exercise lists that are specially adapted to user needs. To attain this outcome, the relevance of the first proposed exercise must be optimized to assist the system in creating personalized user profiles. To solve this problem, this project was designed to include a preliminary testing phase. As a generic tool, AI-VT was designed to be adapted to any field of learning. The most recent application of AI-VT was in the field of computer science specifically in the context of the fundamentals of algorithmic learning. AI-VT can and will also be useful in other disciplines. Developed in Python with the Keras API and the Tensorflow framework, this artificial intelligence-based tool encompasses a supervised learning environment, multi-label text classification techniques and deep neural networks. This paper presents and compares the performance levels of the different models tested on two different data sets in the context of computer programming and algorithms.

Keywords: Algorithmic learning, Deep learning, Personalized learning, Student exercise analysis, Text classification, Natural language processing.

Citation | Carvalho, F., Henriet, J., Greffier, F., Betbeder, M.-L., & Leon-Henri, D. (2023). Deep learning for the detection of acquired and non-acquired skills in students' algorithmic assessments. *Journal of Education and E-Learning Research, 10*(2), 111–118. 10.20448/jeelr.v10i2.4449
History:
Received: 12 August 2022
Revised: 6 January 2023
Accepted: 19 January 2023
Published: 3 February 2023

Publisher: Asian Online Journal Publishing Group

Funding: This research is supported by Fonds Européen de Développement Régional and Fédération de Recherche en Education (Grant number: FC0026398).
Authors' Contributions: All authors contributed equally to the conception and design of the study.
Competing Interests: The authors declare that they have no conflict of interest.
Transparency: The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained.
Ethical: This study followed all ethical practices during writing.

Contents

Contribution of this paper to the literature
This study proposes a system for evaluating students' exercise responses (within the context of algorithmic programming) to determine whether the necessary skills are mastered using deep learning and teacher supervised learning.

1. Introduction

The AI-Virtual Trainer (AI-VT) is a research project that combines skill and sub skill logic with a database to store and assesses data in order to track the user's progress and learning evolution. It is an intelligent, personalized tutoring system based on case-based reasoning (Henriet & Greffier, 2018). The objective of AI-VT is not only assisting teachers in creating and managing their lessons but also to primarily encourage users (in this case student learners) to improve their skills over the course of a training cycle composed of several drill sessions. To achieve this objective, a personalized and carefully balanced list of exercises (in terms of diversity and repetition) is proposed by the system during each session to allow the user an optimal learning experience while avoiding boredom. Each session is thus timed and ends with the attribution of a grade for each of the proposed exercises (this procedure is currently executed in a manual fashion by the teacher). The data is then recorded for future reference. This system was initially designed for sports and in particular aikido but is now used in the field of computer science to teach and review the basic rules related to computer programming, particularly algorithms. Since AI-VT can be used as a teaching and learning tool in a variety of disciplines. A future phase of this project will involve integrating AI-VT into the context of English for non-specialist courses at the undergraduate level.

In order to offer personalized exercises, AI-VT uses information from the user's previous training sessions. However, this system is forced to use default values which implies that this session will initially be the same for all users. For this reason, the AI-VT project is evolving by adding a module. This module is a preliminary test that aims to identify the acquired and non-acquired sub-skills of the user in order to establish a unique user profile. This procedure was previously carried out in a manual fashion by each student by means of the self-evaluation of their subskills for each of their completed exercises. Observations and corroboration by the teacher revealed that students did not effectively assess their exercises which in turn further complicated the overall efficacy of the learning environment since the teacher was obliged to review and control their individual results. This step was indeed very time-consuming. In fact, the aforementioned observation results mirrored the Dunning-Kruger effect which states that the less competent students tend to overestimate themselves whereas the competent students tend to underestimate themselves (Kruger & Dunning, 1999).

The pre-diagnostic system for this AI-VT project was thus created to verify the identification of subskills in a text (in the form of computer programming exercises) with the use of artificial intelligence (AI). Nevertheless, the results obtained with machine learning strongly depend on the data used during the learning phase. The student assessments (in the form of exercises) were transcribed into a digital version of this data. Although the results will improve with continuous learning supported by the system, it is important to note that this study is based on a relatively small amount of data in the form of two data sets of computer programming exercises in two different programming languages: Java and JavaScript (with a total of 185 student exercises). In addition, the comparison of the results between the two exercises by performing AI training per exercise was pertinent to identify the elements of the text that can influence the performance of the system. Future research in this area should include a relatively larger amount of data composed of a larger number of different exercises in order to produce AI capable of identifying the subskills in any simple computer program.

The next few sections of this article present the status of current research in this area as well as a more detailed description of the architecture, research and results of this AI-VT project.

2. Related Work

With the advent of Massive Open Online Courses (MOOCs) which are online courses (often free of charge) composed of large cohorts of students, the presence of AI is required to manage the massive amount of data and high number of enrolled students. Thus, AI has increased in the field of education in order to automate some of the associated processes where classic software has often failed. This kind of system is implemented to reduce the workload of teachers and support learners in their new online learning environment similar to the system presented by Bhatia and Singh (2016) which has been used in the context of edX computer courses (https://www.edx.org/). The system automatically corrects syntax errors in Python functions with the assistance of recurrent neural networks. In addition, Hussain, Zhu, Zhang, Abidi, and Ali (2019) propose a system that is based on student-generated data collected throughout the academic year to predict which students will have difficulty during exam sessions. Some research projects are oriented towards source code processing and focus on the identification and documentation of difficulties by automatically generating comments or code reviews. Based on a Long Short Term Memory (LSTM) process, the Gupta and Sundaresan (2018) system is able to generate code review messages, through pair (code and review message) training for which the review message may or may not be relevant. Additional research projects, Iyer, Konstas, Cheung, and Zettlemoyer (2016); Kannan et al. (2016); Nguyen, Nguyen, Nguyen, and Nguyen (2013) or Singh, Gulwani, and Solar-Lezama (2013) have described methods that are capable of performing similar tasks based on source code processes or text analysis by generating summaries or suggestions. There are also systems with a similar goal to this project. Wang, Su, Ma, Wang and Wang (2011) proposed a research method oriented towards skill acquisition that encompasses the development of learning by considering student skills. However, they have limited their study to static analyzers and dynamic tests and they do not integrate the use of AI. Although these projects are used in a related field, they mainly allow for text generation rather than analysis and do not take deep learning into consideration. With this kind of system, the rules must be pre-defined with the experts whereas the AI-VT approach consists of allowing the system to learn by itself from the furnished data (corrected student exercises).

The data analysis methods most adapted to this project from the perspective of Natural Language Processing (NLP) are text classification. Recent research has involved the use of deep learning for data analysis within the context of media communication and more precisely social networks; Jianqiang, Xiaolin, and Xuejun (2018) focus

on determining whether the sentiments of Twitter (https://twitter.com) posts are of a positive or negative nature. Their analysis is based on a binary text classification process that integrates a global vector for word representation called a Deep Convolutional Neural Network (GloVe-DCNN). This led to the discovery of the multi-class and multi-label text classification methods. The binary classification mentioned above allows for a single output value, the AI-VT project requires a value for each subskill and for this reason, the multi-label classification method was adopted. The data analysis and multi-label methods explained by Agarwal (2020) and Nabi (2018) were adopted to determine if they could be suited and adapted to the AI-VT tool.

3. AI-VT Architecture and Evolution

3.1. AI-VT Architecture

AI-Virtual Trainer is a tutoring system based on Case-Based Reasoning (CBR). Figure 1 presents the architecture of AI-VT as a multi-agent system (MAS). A MAS constitutes a paradigm designed to handle distributed systems. In MAS, an agent is a physical or abstract entity having certain specific characteristics: perception of its environment (including itself and the other agents), capacity to act (upon itself or the environment) and autonomy in its decisions and actions. In AI-VT, the choice of sub-capacities regarding a given capacity takes place through an autonomous process as does the determination of exercises regarding a sub-capacity or of any other exercises chosen and their priority levels. The initial choice of exercises regarding a sub-capacity must be an autonomous process: each agent's autonomy ensures a wise and free selection of the most suitable exercises. These processes can be undertaken simultaneously after the determination of sub-capacities. In addition, each one must interact with the other processes and take their choices into account: the solution proposed by one agent influences the choices made by the others.

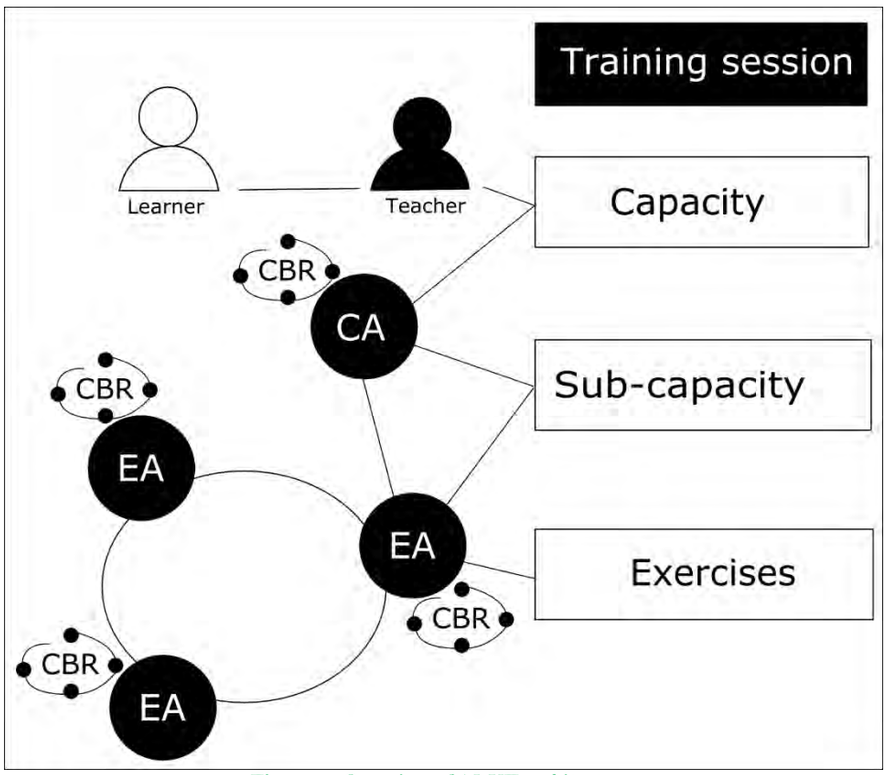

Figure 1. Overview of AI-VT architecture.

As shown in Figure 1, the system is composed of four different agents: the teacher, the learner, the capacity agent (CA) which is responsible for choosing the sub-capacities regarding a capacity requested by the teacher and the exercise agents (EA). Each of these agents plays a part in determining which of the exercises are best suited for the student to improve in each sub-capacity. The CA is directly connected to the exercise agents and can exchange messages. The CA sends the set of sub-capacities it has chosen to one of the exercise agents. This first-contacted exercise agent (EA) endorses the role of coordinator between the CA and the other EAs. This EA assumes responsibility for the first proposed sub-capacity and then creates and sends the list to another EA which assumes the second sub-capacity and so on. The EAs then communicate and share information in order to prepare the requested training session. Each EA proposes exercises concerning its assigned sub-capacity considering the choices proposed by the other EAs. For example, one of the system's requirements is that each exercise be done only once during the entire training session. Thus, the choices of the EAs are shared. Therefore, the referent version of the training session is transmitted from EA to EA until it fulfills all the requirements. Finally, the EA initially contacted by the CA sends the reference version of the training session back to the CA.

3.2. AI-VT Evolution

As described above, in order to have an accurate student profile from the first use of AI-VT, it was necessary to add a module to the existing system. Using a few exercises, this module can define the acquired sub-skills and propose remedial exercises for improvement. This diagnosis process is presented by the AI model illustrated in the next section.

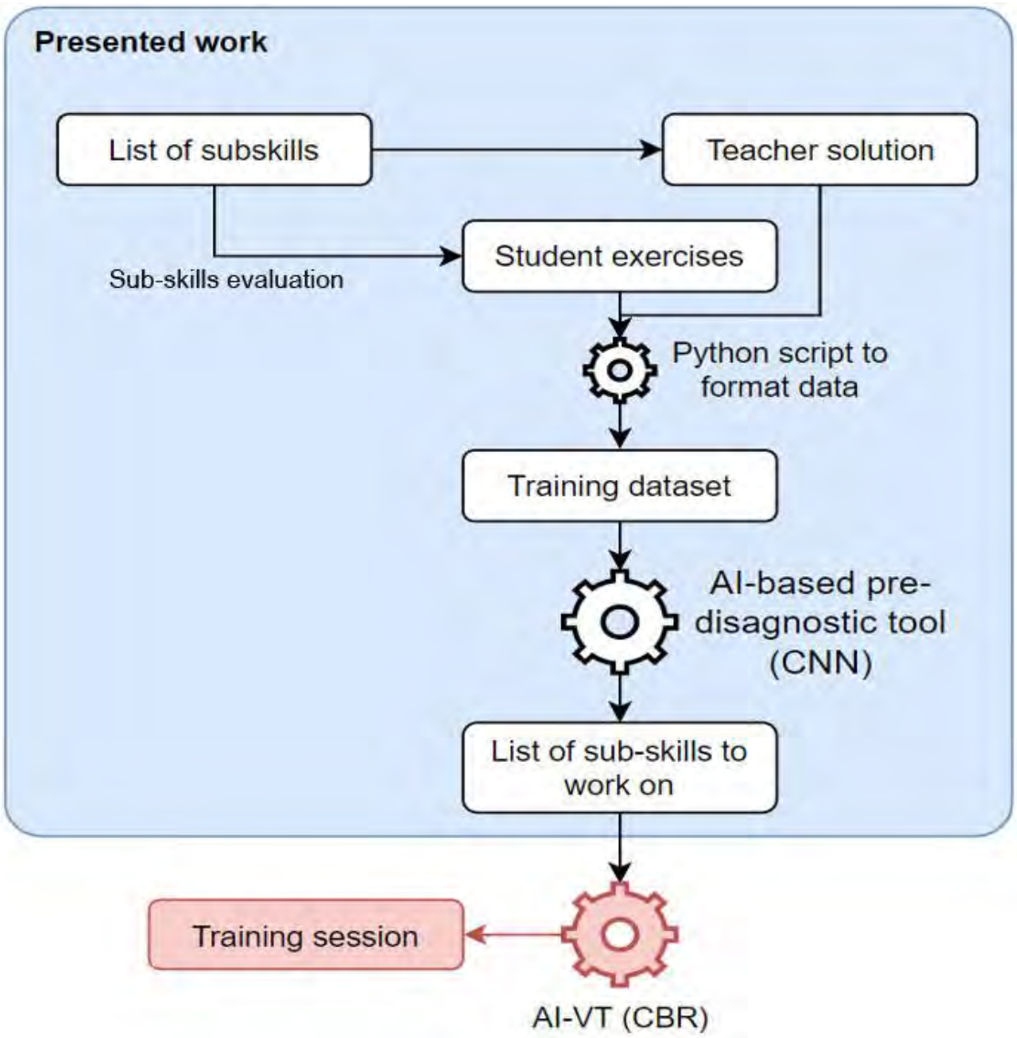

Figure 2. AI-VT pre-diagnosis phase: AI-VT and skill level assessment.

Figure 2 shows the evolution of the AI-VT project from the original version to the one proposed in this article with details on the composition of the training data set used for this study.

4. DCNN for Subskills Definition in Source Code

This tool must be able to perform two tasks: the first is a preliminary phase of data formatting and the second is the analysis phase based on AI for which we must find an appropriate model. Recurrent neural networks appear to be more suitable for generating text while convolutional networks appear to be more suitable for analysis purposes as shown by the previously cited research. Although the two most commonly used types of networks are the CNN (Convolutional Neural Network) and the LSTM (Long Short-Term Memory), only the CNN network was used for the purpose of this research.

4.1. Dataset

In the context of this study, the multi-label text classification method was employed along with supervised learning. Two different data sets were used to train the AI: two sets of digitized student assessments consisting of program code in two different languages (Java and JavaScript) and including an answer to two different exercise statements. In Table 1, the instructions are presented for both programming exercises:

Table 1. Exercise instructions.

Dataset	Exercise
Java	Ask the user for three letters. The program should display a sentence and note whether the letters have been presented in alphabetical order.
JavaScript	Create a *while* or *for* loop in order to fill the first four elements of the *rvb* array with the value 254.

As the two exercises were not proposed to the same classes, they do not offer the same amount of data. Figure 3 shows an example of a student production (or answer) for each exercise:

```
function setup(){
    createCanvas(300, 300);
    i=0;
    while(i<4){
        rvb[i]=254;
        i++;
    }
}
```

(a) Example of JavaScript exercise

```
public class Test1{
    public static void main(String args[]){
        // data declaration
        char letter1, letter2, letter3;

        // get user data
        Screen.display ("Enter a first letter : ");
        letter1=Keyboard.enterChar();
        Screen.display ("Enter a second letter : ");
        letter2=Keyboard.enterChar();
        Screen.display ("Enter a third letter : ");
        letter3=Keyboard.enterChar();

        //processing and display
        if (letter1<=letter2 && letter2<=letter3)
            Screen.display("The letters",letter1,letter2,letter3,
            " are listed in alphabetic order");
        else
            Screen.display("The letters",letter1,letter2,letter3,
            " are not listed in alphabetic order");

        Screen.newLine();
    }
}
```

(b) Example of a Java exercise

Figure 3. Example of a student production for each exercise.

The total number of assessments for each data set is provided in Table 2.

Table 2. Total number of assessments for each programming exercise.

Dataset	Total number of assessments
Java	36
JavaScript	149

The various labels of the multi-label text classification method are presented in this project by the subskills targeted by each exercise. These labels must correspond to those defined in the AI-VT database. The values for each of these labels are 0 or 1 depending on whether the subskill is mastered or not. It is important to note that when the teacher provides the necessary information to generate courses in the AI-VT system, they are allowed to freely select the target skills and subskills on which their students will focus. In the case of this study, both proposed exercises target a subset of the subskills defined by the teacher in the AI-VT system. In the final stage of the preliminary test, each of the subskills must be addressed in at least one of the exercises. Table 3 details the different subskills targeted by each exercise.

Table 3. Targeted subskills for each programming exercise.

Dataset	Target subskills
Java	• Variable concept • User input syntax • Output syntax • Predicate syntax • Predicate logic
JavaScript	• Loop syntax • Variable initialization • Using an index • Defining a step • Defining a stop condition

4.2. Preprocessing Phase

The preprocessing phase consists of data cleaning and the necessary formatting steps on the comma-separated values (CSV) file used for AI training. In the student-produced exercises, it is imperative to remove the items that are not subject to the subskill assessment such as comments, class or method declarations (which are the same in all digitized tests). For the purpose of this study, a simple Python script was used to execute this preprocessing phase. The second step is to format the data so that it can be used with the AI. According to Hindle, Barr, Gabel, Su, and Devanbu (2016), it is possible to perform the first tests with a pre-trained word embedding GloVe. However, this method was quickly abandoned due to initial unsatisfactory results. It was then necessary to find another means to represent the source code. As the program or student exercise is not necessarily syntactically correct, a representation with an abstract syntax tree was discarded. In addition, the system is intended to be generic and in this way of formatting, the code would be inappropriate in the context of other possible field applications. Consequently, the tokenization method (which transforms a text into a sequence of numbers by associating an integer with each of the words) was selected. The digitized assessments are divided into token sequences and then each of them is encoded into an integer according to the number of occurrences. Since most existing tokenizers were designed to process text written in natural language and not computer code, they were not suitable for our case. Therefore, for this study, a unique and innovative solution with Python was necessary for the tokenization phase.

4.3. Model

Since we only have access to a small amount of data, we must use simple models and that's why we used the most common models which are CNNs and LSTMs. These models were constructed with the Keras Application Programming Interface (API) and within the Tensorflow framework. It was impossible to achieve consistent results with the LSTM-based model, the CNN model was chosen. The following section details the results of this method.

Based on three layers of convolutions, the Keras model in Figure 4 illustrates the sequential model that was adopted:

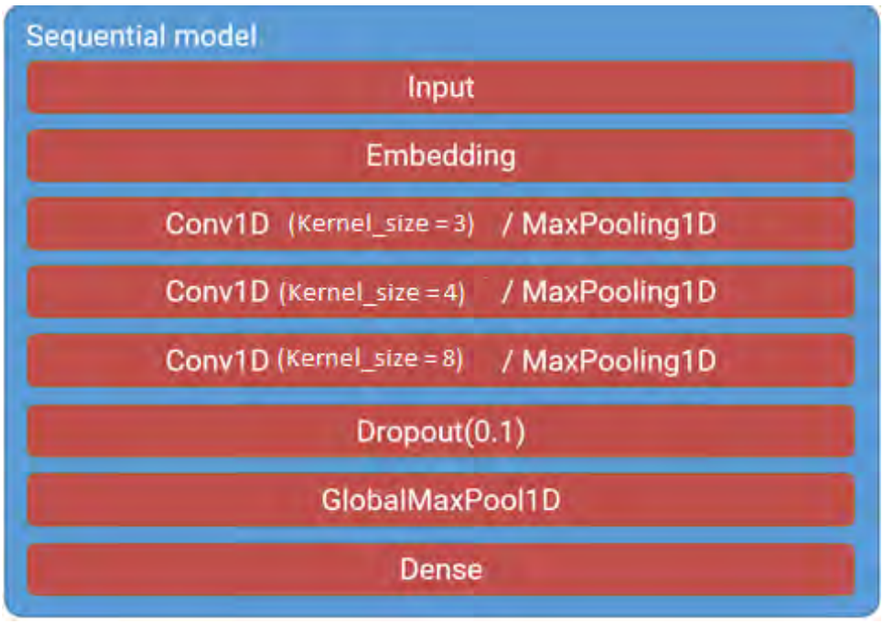

Figure 4. Keras model with different layers.

The usual layers of a text classification are represented with an embedding layer, a dropout layer and a dense output layer. For the convolution layers, three levels of kernels are integrated. Two small kernels (respectively size 3 and 4) are used for the first two to consider short sequences and then a size 8 kernel for the longer sequences.

5. Results

The model's performance evaluation was conducted by means of an NVIDIA GTX 1060 6GB GPU-equipped laptop by using two metrics (precision and false negative rate) and by performing measurement tests on the two data sets. Regarding the rate of accuracy or the proportion of correct predictions, the results are presented in Table 4.

Table 4. Accuracy percentage obtained for the two datasets.

Dataset	Accuracy percentage
Java	70.4
JavaScript	81.6

The accuracy rates obtained for the two data sets are quite high but there is still a significant difference between them. This metric analysis method is used to assess the overall performance of the model. However, it is not used to identify when the AI is erroneous in the analysis phase. Therefore, the false negative rate was implemented.

The false negative rate represents the proportion of predictions for which AI considers that a subskill is acquired when it is not acquired by the student. The results reported in Table 5 are critical because they reveal that the students may potentially unknowingly be blocked by the system and not have the opportunity to work on an unmastered subskill.

Table 5. False negative percentage obtained for the two datasets.

Dataset	False negative percentage
Java	24
JavaScript	22.6

The results in Tables 4 and 5 indicated a marked difference in the accuracy and false negative percentages for the two datasets. In order to reduce this difference but also to globally improve them, we have emitted several hypotheses about the elements that can negatively impact the training phase on the Java dataset but not on the JavaScript dataset:

- The length of the token sequences: the Java exercise and the data set therefore include longer token sequences. Can this size difference play an active role in rendering the task of subskills identification more challenging for AI?
- Strings: The Java exercise requires a user text output and strings. Although there are similarities between assessments, the use of different sentences will result in a significant increase in the number of tokens. For instance, a misspelled word will be considered a new token. Can the number of different tokens perhaps have an impact on the performance of AI learning?

- The amount of furnished data: the most probable hypothesis concerns the amount of data available for each of the exercises. The JavaScript data set contains 149 student assessments compared to only 36 for the Java exercise. Can this difference in data limit the overall quality of AI learning?

To verify the above-mentioned hypotheses, several additional tests were carried out. The standardization of the strings in the student's Java programs led to similar previous results. However, the test that was most successful involved reducing the JavaScript data set to only 36 programs or student exercises in order to have data sets of the exact same size and equal input value. The results of these tests are detailed in Table 6.

Table 6. Accuracy and false negative percentage on reduced JavaScript dataset.

Dataset	Accuracy	False negative percentage
JavaScript	76.9	27.6

Table 6 reveals a significant decrease in the accuracy percentage but also a significant increase in the false negative rate which proves that the amount of furnished data has an impact on the results. The initial result showed a marked difference between the two data sets likely due to a lack of assessment on the part of the Java exercise.

During the research phase for the most suitable model, the use of an LSTM neural network or a pre-trained word embedding model was tested but quickly abandoned due to insufficient and unsatisfactory results.

Additional tests were introduced and further evaluations were done using the two different techniques in order to ensure these conclusions: LSTM neural networks and pre-trained word embedding. The results of these tests are displayed in Table 7.

Table 7. Accuracy and false negative percentage on reduced JavaScript dataset.

Dataset	Accuracy	False negative percentage
LSTM		
Java	70.5	25.1
JavaScript	82.7	21
GloVe		
Java	71.3	25.5
JavaScript	76.3	29.3

By comparing the results of the CNN model with those obtained with the LSTM model, we notice that the values are similar. The LSTM network does not bring particular interest despite the initial assumptions. The LSTM is a recurrent neural network which means that it is more efficient on sequential data as is the case with the computer programs that make up our data sets.

By comparing the results obtained with GloVe to those obtained using a tokenizer, we notice that the use of this technique does not influence the performance of the AI. As expected, the results are even lower than those obtained with a simple tokenizer which remains consistent since GloVe is trained on "classic" texts while we use texts written in a computer programming language.

6. Discussion

This study tends to demonstrate that AI-VT is now capable of identifying subskills in a source code given by a student through supervised learning-driven AI. This approach is based on the multi-label text classification method in which the labels represent subskills (acquired or not) as well as on a CNN.

However, this system has shown certain limitations particularly in terms of the quantity of data (in the form of data sets or student exercises) that was initially furnished. For example, the teacher is not capable of determining whether a skill is acquired or not in the case of answers that are left unanswered by the student. In this case, regardless of the reason for which a student did not provide an answer, the skill is clearly not acquired (and thus must be noted as 1).

An additional limitation arises when a subskill is very difficult to master. This frequently occurs when there are very few excellent student responses or a very low number of programming exercises that contain subskills without any errors. In this context, the AI will not have enough data to efficiently assess and identify the potential errors and inaccuracies that are possible. In more general terms, the recurrent factor that appears to increasingly limit the system is the amount of data that is available for the AI to learn the difference between correct (acceptable) and wrong (inaccurate).

Further research in this area will consist of integrating this pre-diagnostic tool into the AI-VT project and training the AI on a larger number (200 or more) of digitized student exercises or assessments. A larger quantity of data will be used to verify the impact of the amount of data on the analysis results. The objective will also be to continue rendering the AI-VT tool as generic as possible in order to be able to use it with other types of text and in particular English language texts. Further analysis and testing will be conducted by means of the utilization of a combination of CNN and LSTM neural networks in order to improve the efficiency of the AI model.

This research project has also yielded a vast number of improvements and future applications for the AI-VT tool and approach to personalized learning. These include and are not limited to automated real-time decisions regarding the choice of personalized student training exercises. Furthermore, since the system is now capable of determining student subskill capacities from the analysis of computer programming exercises, future research will include the testing of AI-VT in other fields such as (foreign) language learning.

7. Conclusion

The work carried out in this research study involves deep learning and artificial intelligence based on a CNN, using the multi-label text classification method. Although few research projects have successfully extracted

information from text with the assistance of AI, this system is able to identify the subskills acquired (or not) in a simple computer program produced by a student.

Since AI-VT can be used in a variety of disciplines. The long-term objective is to render this deep-learning tool operational for the greatest number of fields possible with the exception that the said discipline is evaluated by means of a text-based source. This study has shown that larger quantities of data would greatly improve the overall results furnished by the AI even if the system's performance has already demonstrated satisfactory results with a lower quantity of data. Therefore, amplified data quantities and continuous learning will improve the overall long-term performance of AI. Future objectives may also include providing substantial data from multiple exercises to perform a single AI training per discipline rather than one per exercise as is currently the case.

With already promising results, further AI-VT research will involve future representations that encompass much more semantic program information as well as a model that considers the text sequences with the use of a Recurrent Neural Network (RNN).

References

Agarwal, R. (2020). Using deep learning for end-to-end multiclass text classification. Retrieved from: https://towardsdatascience.com/using-deep-learning-for-end-to-end-multiclass-text-classification-39b46aecac81.

Bhatia, S., & Singh, R. (2016). Automated correction for syntax errors in programming assignments using recurrent neural networks. *ArXiv preprint arXiv:1603.06129*.

Gupta, A., & Sundaresan, N. (2018). *Intelligent code reviews using deep learning*. Paper presented at the Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'18) Deep Learning Day.

Henriet, J., & Greffier, F. (2018). *AI-VT: An example of CBR that generates a variety of solutions to the same problem* Paper presented at the International Conference on Case-Based Reasoning.

Hindle, A., Barr, E. T., Gabel, M., Su, Z., & Devanbu, P. (2016). On the naturalness of software. *Communications of the ACM*, *59*(5), 122-131.

Hussain, M., Zhu, W., Zhang, W., Abidi, S. M. R., & Ali, S. (2019). Using machine learning to predict student difficulties from learning session data. *Artificial Intelligence Review*, *52*(1), 381–340. https://doi.org/10.1007/s10462-018-9620-8

Iyer, S., Konstas, I., Cheung, A., & Zettlemoyer, L. (2016). *Summarizing source code using a neural attention model*. Paper presented at the Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume Long Papers).

Jianqiang, Z., Xiaolin, G., & Xuejun, Z. (2018). Deep convolution neural networks for twitter sentiment analysis. *IEEE Access*, *6*, 23253-23260. https://doi.org/10.1109/access.2017.2776930

Kannan, A., Kurach, K., Ravi, S., Kaufmann, T., Tomkins, A., Miklos, B., . . . Young, P. (2016). *Smart reply: Automated response suggestion for email*. Paper presented at the Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

Kruger, J., & Dunning, D. (1999). Unskilled and unaware of it: How difficulties in recognizing one's own incompetence lead to inflated self-assessments. *Journal of Personality and Social Psychology*, *77*(6), 1121-1134. https://doi.org/10.1037/0022-3514.77.6.1121

Nabi, J. (2018). Machine learning - word embedding & sentiment classification using Keras. Retrieved from: https://towardsdatascience.com/machine-learning-word-embedding-sentiment-classification-using-keras-b83c28087456.

Nguyen, T. T., Nguyen, A. T., Nguyen, H. A., & Nguyen, T. N. (2013). *A statistical semantic language model for source code*. Paper presented at the Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering.

Singh, R., Gulwani, S., & Solar-Lezama, A. (2013). *Automated feedback generation for introductory programming assignments*. Paper presented at the Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation.

Wang, T., Su, X., Ma, P., Wang, Y., & Wang, K. (2011). Ability-training-oriented automated assessment in introductory programming course. *Computers & Education*, *56*(1), 220-226. https://doi.org/10.1016/j.compedu.2010.08.003