future internet

MDPI

Article

Real-World Implementation and Integration of an Automatic Scoring System for Workplace Safety Courses in Italian †

Nicola Arici [1,2,*], Alfonso Emilio Gerevini [1], Matteo Olivato [1], Luca Putelli [1], Luca Sigalini [2] and Ivan Serina [1,*]

[1] Department of Information Engineering, University of Brescia, Via Branze 38, 25121 Brescia, Italy; alfonso.gerevini@unibs.it (A.E.G.); m.olivato@unibs.it (M.O.); luca.putelli@unibs.it (L.P.)
[2] Mega Italia Media, Via Roncadelle 70A, 25030 Castel Mella, Italy; luca.sigalini@megaitaliamedia.it
* Correspondence: nicola.arici@unibs.it (N.A.); ivan.serina@unibs.it (I.S.)
† This paper is an extended version of our paper published in "A BERT-based Scoring System for Workplace Safety Courses in Italian", AIxIA 2022—Advances in Artificial Intelligence—XXIst International Conference of the Italian Association for Artificial Intelligence, AIxIA 2022, Udine, Italy, 28 November–2 December 2022.

Abstract: Artificial Intelligence and Natural Language Processing techniques can have a very significant impact on the e-learning sector, with the introduction of chatbots, automatic correctors, or scoring systems. However, integrating such technologies into the business environment in an effective way is not a trivial operation, and it not only requires realising a model with good predictive performance, but also it requires the following: (i) a proper study of the task, (ii) a data collection process, (iii) a real-world evaluation of its utility. Moreover, it is also very important to build an entire IT infrastructure that connects the AI system with the company database, with the human employees, the users, etc. In this work, we present a real-world system, based on the state-of-the-art BERT model, which implements an automatic scoring system for open-ended questions written in Italian. More specifically, these questions pertain to the workplace safety courses which every worker must attend by law, often via e-learning platforms such as the one offered by Mega Italia Media. This article describes how our system has been designed, evaluated, and finally deployed for commercial use with complete integration with the other services provided by the company.

Keywords: natural language processing; System Integration; deep learning

Citation: Arici, N.; Gerevini, A.E.; Olivato, M.; Putelli, L.; Sigalini, L.; Serina, I. Real-World Implementation and Integration of an Automatic Scoring System for Workplace Safety Courses in Italian. *Future Internet* **2023**, *15*, 268. https://doi.org/10.3390/fi15080268

Academic Editor: Michael Sheng

Received: 5 July 2023
Revised: 1 August 2023
Accepted: 9 August 2023
Published: 12 August 2023

1. Introduction

In recent years, the e-learning industry has been subject to exponential growth due to several factors, such as the COVID-19 pandemic, the development of new technologies, and the increasing demand for learning that traditional "classroom" methods could not satisfy, especially in environments such as the workplace. In particular, millions of workers, depending on the industrial sector and the country in which they work, are required by law to take workplace safety courses every year. In Italy, every new worker has to attend a course of at least eight hours relating to a general workplace security course and a more specific course for his/her sector. Obviously, riskier jobs require more training, and all employees after a certain number of years must take a refresher course.

Mega Italia Media is a leading company in the e-learning sector with a specific focus on occupational safety training. In the last 30 years, this company has helped organisations and professionals train millions of workers in health and safety at work. In 2011, they released their e-learning platform, DynDevice (https://www.dyndevice.com (accessed on 8 August 2023)), a powerful tool that facilitates companies in standard corporate training and allows them to create, if necessary, specific courses for their employees.

Due to their mandatory nature, every course provided by Mega Italia Media ends with an assessment test to determine if the worker has acquired the skills required in the course. Typically, these tests consist of either multiple-choice questions, where a user has to choose the answer in a set of predetermined options, and open-ended ones, where a user

has to write a complete answer. Every day, Mega Italia Media's users compile hundreds of tests; while the first type of tests can be corrected automatically and do not require human contribution for evaluation, this is not the case for open-ended question tests. In fact, these are corrected by human experts who have to dedicate one or two hours every day for this routine operation. Therefore, it is certainly useful to build an automatic corrector that uses artificial intelligence to reduce the workload of human operators.

In this paper, we describe our scoring system to address this issue, and the main focus of our work is explaining how this kind of technology can be best integrated into the real-world environment and not only in a testing ground, as done in a previous work [1].

In order to do so, we first analysed the case study and identified the critical points where artificial intelligence can assist industry experts. At this point, we performed a theoretical and practical analysis of the different types of courses, tests, and user case studies that the company tackles every day. Thus, to solve our task, after analysing the literature in this field, we decided to use the BERT [2] Transformer-based model [3], a state-of-the-art approach for many natural language processing tasks.

Once the model performed in an acceptable way in a controlled environment, we deployed the deep learning prototype in a fully operating model and integrated our artificial intelligence solution into the company workflow and software. Finally, assisted by domain experts, we performed a real-word evaluation on the actual tests submitted by hundreds of users.

Despite the application of BERT being basically the state-of-the-art for many NLP tasks [4–6], the main contribution and final goal of this project is to show how a complete artificial intelligence system can be created to assist a company in the daily, hard, and time-consuming task of correcting tests. Therefore, we describe a software architecture which has been successfully integrated into an already existing platform, such as the one by Mega Italia Media, and not only the training and evaluation of a BERT model. In this work, our main contributions (also with respect to our previous paper [1]) are:

- we conduct a more in-depth analysis and experimentation, considering different BERT models and different input configurations;
- we give a more detailed description of our case study and our datasets;
- we describe the entirely new real-world implementation of our system, its architecture, its main components, and how they interact with one another;
- we measure the performance of our system in this real-world context, providing a qualitative and quantitative analysis of its behavior and its impact on human evaluators and students.

The structure of this paper is as follows: in Section 2, we review related works; in Section 3, we present the core of our deep learning model, BERT; in Sections 4 and 5, we describe our case study and the dataset used to train and evaluate the models; in Section 6, we explain the architectures of the models and the fine-tuning operation, and in Section 7 we discuss the experimental results; in Section 8, our real-world implementation of the models is presented; and, finally, Section 9 contains our conclusions and hints at future work.

2. Related Work

Natural Language Processing (NLP) techniques can be applied in different ways and to different real-world and industrial cases. Many NLP techniques are also implemented in machine learning or deep learning applications to solve specific tasks in specific domains. In the healthcare system, NLP is used to automatically classify clinical reports [7] or to infer people's mental states from what they write on social media [8]; in the marketing and customer care sectors, several companies implement an NLP application to predict customer needs [9] or analyse customer reaction or satisfaction [10]. These models typically use reviews or social media posts and tweets to predict how users feel about a specific topic, and companies can use this information to improve their marketing strategies.

A very popular and implemented NLP solution consists of conversational systems. In recent years, many companies have created their virtual assistant solutions to improve customer communication, provide better 24/7 support, and reduce the workload of employees [11,12]. A conversational system is a complex structure that leverages on several NLP subsystems, such as a Natural Language Understanding model and a multi-label classifier that predicts the intent of the user. In addition to NLP components, the realisation of a complete chatbot requires other structures, such as a front-end interface and a back-end data layer [12]. Today, there are many different frameworks that provide this kind of service, such as Watson IBM, Google Dialogflow, Microsoft Luis, and Facebook Wit.ai [13].

In recent years, several approaches and frameworks have been developed to solve scoring and graduation tasks. The two main branches of this problem are: Automatic Essay Scoring (AES) [14] and Automatic Short Answer Grading (ASAG) [15]. This second task can be tackled in two ways: with a regression approach, which tries to predict a score in a continuous range (usually between 0 and 5), and a classification approach, in which the system has to assert the correctness of the user's answer.

Following the evolution of machine learning and deep learning, over the years, many different techniques have been implemented to solve ASAG tasks. One of the first methods combines the Bag-of-Words representation and the Support Vector Machine to provide a score [16]. The words contained in the sentences were also used with overlapping algorithms to measure the similarity between the reference and the student's response [17]. Similarly, Gomaa and Fahmy [18] proposed an unsupervised Bag-of-Words approach that deals with students' responses using text-to-text similarity, in particular 13 string-based and 2 corpus-based similarity measures combined to compute the similarity. Another popular method that uses the words contained in the answers is presented in [19]; its authors propose a mixed unsupervised and supervised approach that combines a clustering algorithm based on the vocabulary used in the answers and a mathematically supervised model based on the Hamming distance between the reference and the student's answer. All the methods previously described rely on the classical Bag-of-Words NLP technique for representing sentences and documents; however, since the introduction of word embedding techniques [20], Bag-of-Words cannot be seen as the state-of-the-art anymore [21,22]. Moreover, these methods are based on classical machine learning algorithms, while our work proposes a neural network-based approach which allows one to obtain much better performances for text classification [2,7,23] and similarity-based tasks [24,25].

The introduction of word embedding methods such as GloVe [26] and Word2Vec [20] raised the issues of computing a vectorial representation of sentences and how to measure the distance (in terms of their meaning) among them. This task is typically called *semantic similarity*, and the closer the vector representations, the more sentences have a similar meaning. These techniques have been exploited in ASAG by Hassan et al. [27], who proposed a framework based on deep learning word-embedding algorithms to produce a vectorial representation for the student's answer and for the correct reference answer; then, the cosine similarity measure of the two sentences is given to a Ridge regression model to calculate a score for the answer provided by the student.

Another work that leverages the use of embedding methods is Ans2Vec [28], which computes the similarity scores of sentences using the Skip-thought vectors [29]; in particular, this algorithm computes a vectorial sentence representation using an encoder–decoder model. Both the student and the reference answers are converted into Skip-thought vectors, on which features such as component-wise product and absolute difference are calculated; these features are then passed to a Logistic Linear Classifier to compute the student's score. The main drawback of this approach and the one in [27] is that they both use static word embeddings, i.e., representations that do not consider the different meanings into which the words of the sentences are embedded. However, it has been established that dynamic word embedding (which also considers the entire context of the document and the sentence in which a word appears) can achieve better performance [2]. Therefore, our method exploits the state-of-the-art dynamic embeddings generated by BERT.

With the increasing popularity of recurrent neural networks (RNNs), and in particular of Long–Short Term Memory (LSTM) [30], many studies have been devoted to their use for ASAG, in both classification and regression tasks. These technologies exploit the word representations produced by word-embedding algorithms such as Word2Vec and GloVe. Next, they process them to calculate a unique sentence representation that is used to calculate a more accurate score. Prabhudesai and Duong [31] proposed a framework that combined the use of a Siamese bidirectional LSTM. The input of this model is made by the GloVe word vectors of the student and the reference answers, plus a set of handcrafted features, such as the number of total and unique words contained in the students' answers or the ratio between the lengths of the two answers. All these data are concatenated, and a final feed-forward layer produces the score. A similar approach, which also exploits pooling layers, was presented by Kumar et al. [32]. The most important difference from our work methodologically by speaking is that we do not use any Recurrent Neural Network or pre-trained static word embeddings. Instead, we use BERT, which is a pre-trained encoder for dynamic word embeddings, which generally obtains better performance with respect to LSTM-based approaches [2]. In fact, we conducted some preliminary experiments on using LSTM techniques in our context, but we did not obtain satisfactory results.

The real breakthrough in the NLP field came in 2017 with the release of Transformer [3], a new architecture based on the concept of self-attention. Although these models are trained with general documents and tasks, they can be easily adapted to more particular domains by adding domain-related texts in the training phase and can be specialised to solve a specific task through fine-tuning, outperforming (as we stated previously) RNN and LSTM-based approaches.

Sung et al. [33] presented an architecture where a pre-trained BERT model was fine-tuned to solve ASAG tasks on different textbooks and domains. They fed both the student and the reference answers to the language model and they performed a three classes (correct, partially correct, and incorrect) classification with a simple feed-forward layer. Moreover, in this work, the authors showed that adapting a pre-trained BERT with domain-related resources can improve the performance on domain-related questions. However, this procedure has a negative impact on the other domains. A similar procedure was implemented by Lun [34], but in this case, they performed data augmentation on the dataset considered using back-translation and generating different combinations of questions, user's answers, and reference answers. Our automatic correction system is also based on BERT and the use of a simple classifier; however, unlike the works mentioned previously, we consider and evaluate the performance of our system depending on different input configurations, i.e., not using only the model-response–user-response coupling. Moreover, we also provide an overall description of the actual implementation of our artificial intelligent system and how it interacts with human evaluators, the users, and the other components of the I.T. system of Mega Italia Media.

Del Gobbo et al. [35] developed a new framework, *GradeAid*, to solve the ASAG task in different languages. This system first translates the text into English, if needed. Next, it computes the standard TF-IDF matrix and the similarity scores between each student and reference answers with their BERT cross encoder. These two features are then fed into a regressor layer to compute the final student's score. Our system differs from these works in that it was specifically designed and developed for the Italian language (without any translation). Moreover, while all the others evaluate their models on benchmark datasets designed specifically for these tasks, our work is more application oriented and it involves a real-world dataset which is taken directly from what the users actually wrote without any particular refining or pre-processing. Finally, Del Gobbo et al. [35] simply showed the performance of their model without any in-depth analysis on the real-world implementation of the prototypes proposed.

3. BERT

BERT (Bidirectional Encoder Representations from Transformers) [2] is a deep learning architecture based on Transformer [3]. At a very high level, a Transformer is composed of two components: a stack of encoders and a stack of decoders. The first has the role of extrapolating the core information from a text, whereas the latter has the task of producing the output from that piece of information. Since BERT is a model for generating the vectorial representations of words, the decoder stack is discarded and only encoders are used in BERT. The number of encoders that compose the stack is customisable, but the original paper implemented 12 encoders.

As depicted in Figure 1, the model does not accept a plain text sentence in input, but rather an embedding matrix specifically constructed as follows: first, the tokenizer splits a sentence (or a couple of sentences) into tokens, which are full words or parts of words, based on the vocabulary of the model. In output, the tokenizer produces the list of indexes of the vocabulary tokens, and if two sentences are given in input, it also provides a list that specifies which sentence each token belongs to. The tokenizer also plays the key role of adding the special tokens to the sequence; the token [CLS] is always added at the beginning of each sequence, and the token [SEP] is always added between the tokens of two different sentences and at the end of the sequence. Other special tokens, not always added, are the token [MASK], used to substitute a token in special tasks, such as masked language modeling, [UNK], used when the tokenizer does not know how to separate a portion of the sentence, and [PAD], used in short sentences to fill the sequence.

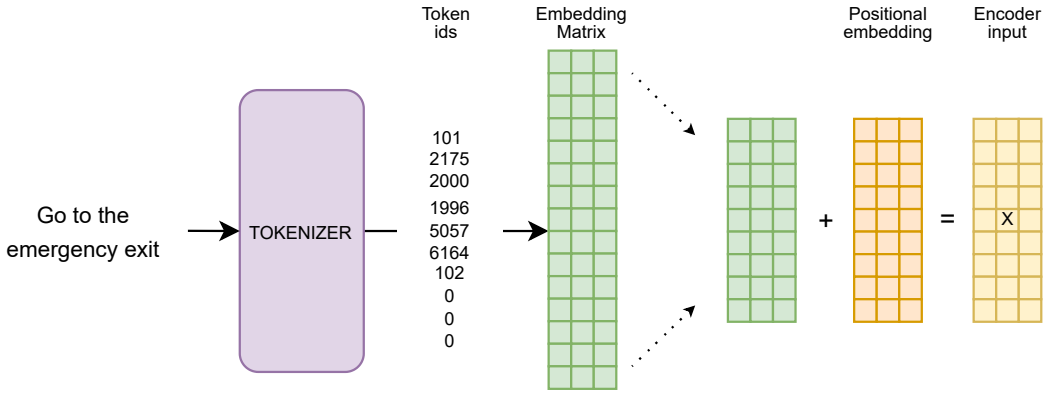

Figure 1. Pre-processing of the textual input for the BERT model. First, the tokenizer splits the sentence into tokens and adds the special tokens required. From the embedding matrix, the embedding layer extrapolates the vector representations of the tokens, in green, and sums the positional embeddings, in red, to produce the input for the model, in yellow.

After the tokenization procedure, the corresponding embedding matrix is built and positional embeddings are added to this representation. These vectors follow a specific pattern that the model learns, which helps it to determine the position of each word in a sentence or the distance between different words in the sequence. It is important to note that the positional embedding depends only on the position of the token in the sequence; therefore, different tokens, in different sentences, in the same index, have the same positional embedding. The output of this operation is the input of the first encoder of the model.

Each encoder, as shown in Figure 2, is made up of two main components: a multi-head self-attention and a feed-forward network. The first one is the core of the model and is composed of a certain amount of independent self-attention mechanisms called *heads* (12 in the original paper). Each head takes as input the same matrix, X, and computes three different vector representations. These are typically called Key (K), Query (Q), and Value (V), and they are computed by multiplying X by three different weight matrices, W_k, W_q, and W_v:

$$K = X \times W_k,\ Q = X \times W_q,\ V = X \times W_v \tag{1}$$

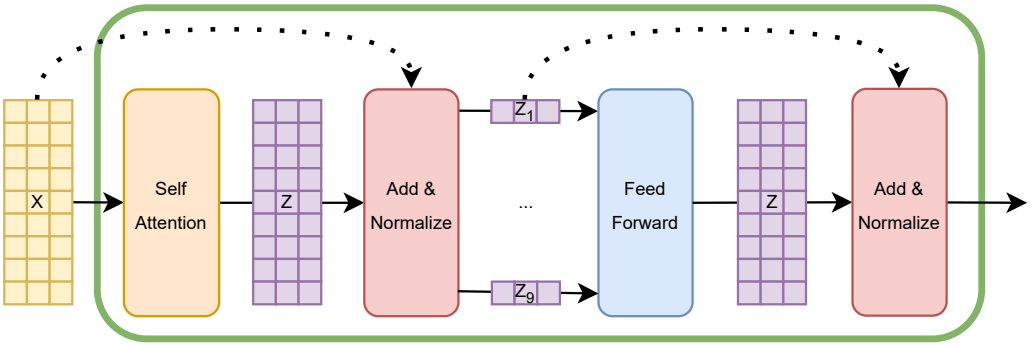

Figure 2. Schematic representation of a BERT encoding layer. The input vectors, X, are processed by a self-attention mechanism, by two add and normalize layers and a feed-forward neural network. The encoder also contains two skip connections from X to the first add and normalize layer and from its output to the second add and normalize layer.

Using these new representations, the self-attention mechanism calculates the attention weights (A) by applying the softmax function on the scaled dot product (i.e., the dot product is divided by the length of the input representation of each token (d_k) between the Query, Q, and the Key, K. Finally, the new representation of each token (Z) is calculated by multiplying the attention weights for V.

$$A = \text{softmax}\left(\frac{Q \times K^{\intercal}}{\sqrt{d_k}}\right)$$
$$Z = A \times V$$

In order to create a single representation provided by the multi-head attention mechanism, the result of each head is concatenated and then multiplied by the weight matrix, W_o, before it is passed to the feed-forward layer.

The feed-forward network is composed of two layers; the first layer expands the d_k feature vectors to the value of the hidden layer dimension value (equal to 3072 in the original paper), whereas the second layer condenses the vector back into d_k features. The same exact network is applied to each token independently and its role is to process the output of an attention layer to better fit the input for the next layer. Finally, each layer of the encoder is followed by a residual connection that adds the output of the layer to its input before a normalisation and a dropout operation are applied.

By using a large corpus of texts (such as newspaper articles, web pages, and other types of documents), BERT is trained using two tasks: *masked language modeling* and *next sentence prediction*. The former consists of learning to identify missing tokens from context. These tokens are called *masked* and typically they are approximately 15% of the corpus. The latter task can be described as follows. Given a sequence made of two sentences, the model learns how to predict whether the second sentence follows the first one in the original document. These tasks are structured this way to allow BERT to understand the overall structure of the language, capturing the most meaningful concepts contained in each word and how these are related to each other in the entire sentence. Once BERT is trained, it can be adapted for specific NLP tasks, such as Named Entity Recognition, Text Classification, or Sentiment Analysis, adding a small, simple layer as a classifier on top of BERT. The smaller and application-oriented datasets are used to train the classifier and to fine-tune the BERT parameters. In our paper, we adapt a pre-trained Italian version of BERT and fine-tune it to solve the short answer grading task on the data that the company had gathered in the past years.

4. Case Study

The Italian company Mega Italia Media is the owner of the e-learning platform DynDevice. This software helps companies, mainly in Italy but also in the rest of Europe, to

provide business training to their employees. Since 2011, Mega Italia Media has opened more than 300 platforms and trained approximately 800,000 workers, mostly through courses on workplace health and safety.

At the end of the course, the worker has to take an assessment test to determine whether he or she has understood the concepts presented in the courses. These tests are composed of two kinds of questions: *multiple-choice* and *open-ended* ones. Although an automatic algorithm is already used to process and correct a multiple-choice question, human intervention is necessary to check open-ended questions.

As stated by the domain experts who perform the correction of the open-ended question-based tests, it takes 10 to 15 min to correct one; therefore, if the number of these tests is particularly high, it will require a very significant effort from the experts. Moreover, the latter are also involved in other duties, such as providing real-time assistance to the users on many platforms.

This issue has become more prominent over the past three years; in fact, as we can see in Figure 3, the number of tests has significantly increased over the years, from 123 in 2012, the year when this service was launched, to 1430 in 2014, up to more than 5000 in 2020 and 6570 last year. Please note that these numbers also include the multiple-choice questions, which are the majority. This growth in numbers is probably due to the major increase in growth of the e-learning sector during the COVID-19 pandemic [36–38]. Therefore, providing a system to help domain experts and speed up the correction process has become a company priority.

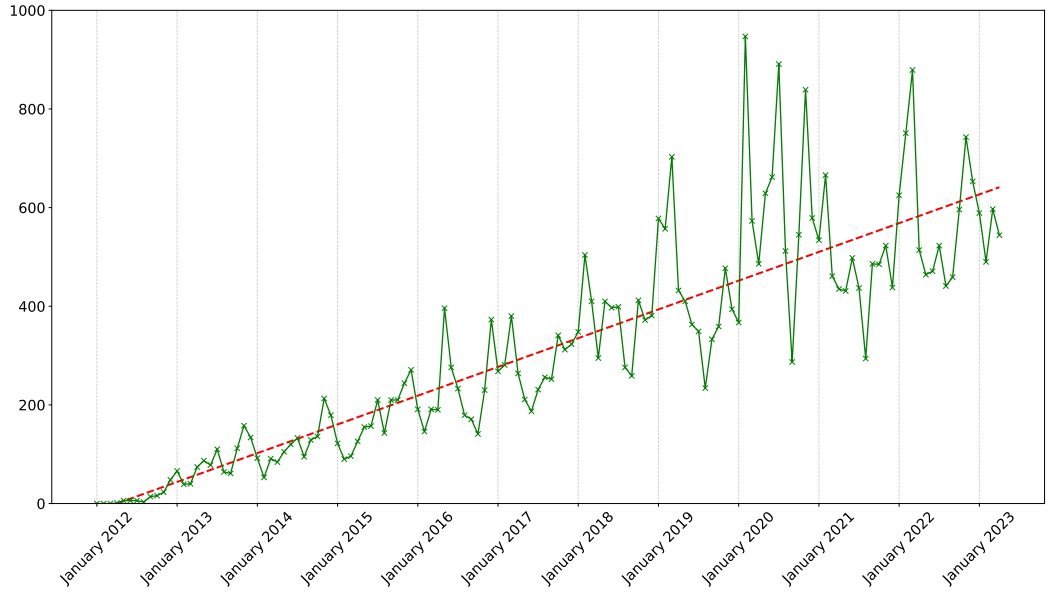

Figure 3. Number of tests from January 2012 to January 2023 (in green). The red dashed line represents the distribution trend, which is a linear increase over the last decade.

Each test is on average composed of five questions that require different kind of answers. Some questions require a short yes or no answer with a brief motivation, others ask for a simple definition, or require articulate thinking on a general topic. In the following, we show some examples of the original questions in Italian. For clarity, we also provide a translation.

1. *Qual è la definizione di lavoro elettrico?*
   What is the definition of electrical work?
2. *Spiegare le differenze tra "lavoro sotto tensione", "lavoro fuori tensione" e "lavoro in prossimità" come definiti dalla Norma CEI 11-27 in relazione all'esecuzione di lavori elettrici.*
   Explain the differences between "live work", "off-voltage work", and "proximity work", as defined in CEI 11-27 in relation to the performance of electrical work.

3. *Quali sono i DPI necessari per eseguire un lavoro sotto tensione (bassa tensione)? E per un lavoro in prossimità (bassa tensione)?*
   What PPE is required for performing live work (low voltage)? What about for proximity work (low voltage)?

For each question, the teacher or the creator of the course also provides a *reference answer* to the experts in the domain to speed up the correction process; this sentence contains all the information that users' answers have to include in order to achieve the maximum score. However, a reference answer can be very complex and long, slowing the correction process. In fact, for domain experts, it is quite difficult to compare the two answers and figure out whether the users wrote all the important information.

Similar issues are also present in automatic correction systems; in fact, this can be seen as a semantic similarity task, a regression-style task that aims to compute a similarity score between two sentences. This operation is not trivial for two long and complex sentences that differ in accessory words (such as conjunctions, adverbs, etc.), and the score produced could not be accurate enough, even with very sophisticated algorithms such as BERT [24,39,40].

5. Dataset

In this section, we present the dataset used to train our artificial intelligence models. The data were extracted from the Mega Italia Media database, a huge repository populated in the last years. This table contains more than 45,000 questions, of which more than a thousand are open-ended questions and with a compiled reference answer. However, by removing duplicate questions, we can see that only 378 unique questions were created by the teachers. For the users' answers, the database contains approximately two million records, but only approximately 135 thousands are answers to open-ended questions. Even in this case, once we remove the duplicate records and the answers with no score, the number of the users' answers is reduced to 7315.

In Table 1, we present a simple analysis of the length of the different data; as we can see, the questions are several words shorter than the answers. Moreover, it is important to note that, on average, the user's responses have fewer words than the reference answers. This is probably due to the fact that the teacher that creates the course puts all the information that a user response can contain in these sentences, but the users manage to express the same important concepts with fewer words and more concisely. This relevant difference in length can harm the automatic correction system and, as stated by the domain experts, it is difficult even for them to correct a test with these long sentences. Thus, in order to facilitate the correction process and improve the performance of the scoring system, we asked the domain experts to extract a set of *key concepts* from the reference answers, that are not just key words, but simple sentences, typically 5 to 10 words long.

Table 1. Percentile analysis of the length of the questions, reference answers, and user answers in the company database. In the table, the 1st; the 99th; and the value of the 25th (first quartile), 50th (the median), and the 75th (third quartile) are reported.

Type	1st	25th	50th	75th	99th
Question	8	20	26	34	55
Reference Ans.	23	42	67	97	275
User Ans.	4	25	45	79	283

With the introduction of the key concepts, we shift from a classic regression task, when the system has to compare the two answers and give a score, to a classification task; we ask the deep learning models to classify the presence of the concept in the user answers, and predict 1 if the concept is present in the user answer, and 0 otherwise. We derive the label from the user answer score that the domain expert assigns in the correction process; we assign label 1 if the score is above a certain threshold, 0 otherwise. For consistency, even

in the case where our models compared the two complete responses, we asked to rank whether the user answer score exceeded the threshold mentioned above.

The real problem of this approach arises due to the strong imbalance of the score distribution. At least 75% of the scores are greater than or equal to 80 out of 100, with 25% that obtained the maximum score. If we chose the standard Italian threshold, 60 out of 100, we would have only 7.35% of the data with label 0; thus, to reduce the class imbalance, we increase the threshold to 80 and obtain a 28.49% of the data with 0 class label. This choice is also justified by the fact that our deep learning models are part of a semi-supervised automatic corrector, and all the predictions must be validated by a domain expert for legal reasons. So, with the threshold set at 80, if the model prediction is 1, the expert has a high confidence that the concept is present, whereas if the prediction is 0, the expert has to control more carefully the real presence of the concept. Given the strong imbalance towards high scores, it is fair to expect that the number of concepts actually present will exceed the others that are absent; as a consequence, this architecture, which makes it easier correcting the concepts predicted as present than the ones predicted as absent, greatly speeds up the correction of tests.

Unfortunately, due to the short time and large amount of work, the domain experts were only able to extract 114 concepts from 67 questions, and so to create a standard benchmark across models, we had to discard more than 300 questions without extracted concepts. From these 67 questions we retrieved 4117 user responses that we used to build our dataset. Each instance is formed by the question posed to the user, the old complete reference answer and the user's answer, a single concept connected to the latter, and the score given to the answer. If the answer had more than one concept, we duplicated the entire record and only changed the concept field. We obtained 10,560 records in this way.

In order to train, validate, and test our deep learning models, each dataset was divided into three parts: 80% for training, 10% for validation, and 10% for testing.

6. Models

So as to build the core artificial intelligence of our application, we implement BERT (Bidirectional Encoder Representation from Transformers), an only encoder Transformer model. To solve our task, we build our classification model by adding a simple classifier to BERT. As shown in Figure 4, the latter consists of a simple linear feed-forward layer preceded by a dropout operation.

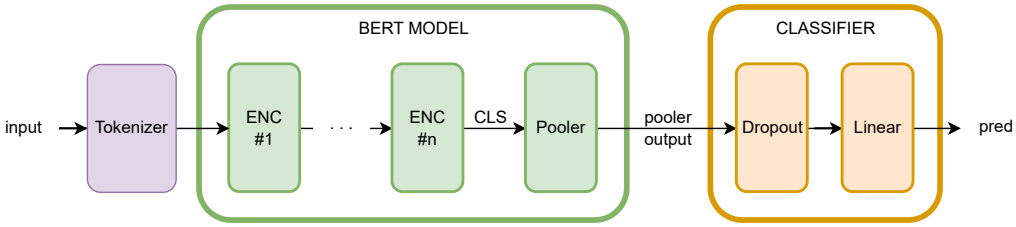

Figure 4. Artificial intelligence architecture. First, the textual input is tokenized and then the tokens are processed by the BERT model to produce the sentence representation; finally the classifier, composed of a dropout and a simple feed forward layer, produces the binary prediction.

Starting from this architecture, we build four types of models that differ only in the input. First, we create the *Reference Model* (REF), where we concatenate the user's response with the reference answer. Then, in order to improve the performance, we want to give BERT more context and add the question to the input, and in this way we obtain the *Question-Reference Model* (QREF); in these cases, the models are trained to predict if the response has a score greater than 80. We follow the same procedure with the concepts; first we create the *Concept Model* (CONC) to solve the task by matching the user's answer with a single concept, and then, with the *Question-Concept Model* (QCONC), we add the question as well. In this case, the goal is to predict whether the concept is contained in the user response or not. For clarification, we show some examples of input below:

1. **REF**: Go to the emergency exit [SEP] Crawl on the floor and proceed towards the emergency exit.
2. **QREF**: What to do in case of fire? [SEP] Go to the emergency exit [SEP] Crawl on the floor and proceed towards the emergency exit.
3. **CONC**: Go towards the emergency exit [SEP] Proceed towards the emergency exit.
4. **QCONC**: What to do in case of fire? [SEP] Go towards the emergency exit [SEP] Proceed towards the emergency exit.

Each configuration is fine-tuned using the same procedure. We choose the Binary Cross Entropy loss function and we use the AdamW algorithm as the optimizer. Moreover, we pick a linear scheduler and set the number of warm-up steps equal to 10% of the training iterations as the learning rate scheduler. Given the strong class imbalance, each model is validated using the F1 score, a more precise metric in this case than accuracy. In the next section, we report in detail all the experiments performed to obtain the best model to deploy.

7. Model Selection and Prototyping

In this section, we present the experimental results obtained in the different models and fine-tuning configurations; the best version of each model obtained in each trial is validated on the test dataset, whose composition is shown in Table 2. In the early stages of our project, we tried an LSTM-based approach and pre-trained Italian word embeddings, but we did not obtain sufficient or promising results.

Table 2. Test dataset composition. In the first row, the data are aggregated by answers and in the second by concepts.

Type	# Instances	# 0 Class	# 1 Class	% 0 Class
Answers	412	282	130	31.55%
Concepts	1065	717	348	32.68%

7.1. Experimental Setting

As a first experiment, we trained our models with a fixed set of hyperparameters over 50 epochs. Given that these models reach a plateau on the 0-F1 metric after 8–10 epochs, we decided to keep the hyperparameters involved in the fine-tuning operation fixed, such as the optimizer learning rate equal to 2×10^{-5}, its weight decay equal to 0.01, and the batch size equal to 16. We made these choices because a small modification of these hyperparameters did not significantly impact the results for a fine-tuning of only 10 epochs.

Each model was evaluated on the test set on the F1-score metric, calculated as the harmonic mean of precision and recall; given the high imbalance in the class labels, we focused on maximising the F1 score for the zero class (0-F1).

First, we decided to test different types of pre-trained BERT models that are freely available on the HuggingFace repository (https://huggingface.co/ (accessed on 8 August 2023)); we tested four configurations of an Italian BERT model (https://huggingface.co/dbmdz/bert-base-italian-uncased; https://huggingface.co/dbmdz/bert-base-italian-cased; https://huggingface.co/dbmdz/bert-base-italian-xxl-uncased, https://huggingface.co/dbmdz/bert-base-italian-xxl-cased (accessed on 8 August 2023)): cased or uncased, base or XXL.

The cased version, unlike the uncased one, recognizes capital letters and normal letters. Moreover, the base model version is trained with the Italian Wikipedia dump (approximately 13GB of text), whereas the XXL version is trained with the Italian part of the OSCAR corpus (in total approximately 81GB of training data). Furthermore, we tested the multilingual cased and uncased versions of BERT trained by the authors of the original paper (https://huggingface.co/bert-base-multilingual-uncased; https://huggingface.co/bert-base-multilingual-cased (accessed on 8 August 2023)); we then tried two different models, a multilingual RoBERTa (https://huggingface.co/xlm-roberta-base (accessed on

11 August 2023)) [41] and an Italian Electra (https://huggingface.co/dbmdz/electra-base-italian-xxl-cased-discriminator (accessed on 11 August 2023)) [42].

These two models share with BERT the same base architecture, but they have a different pre-training procedure; RoBERTa is trained only on the Masked Language Model task, where Electra is trained with a generator-discriminator approach on a variation of the MLM task. In this second case, given a sentence with a [MASK] token, the generator model has the goal of filling the sentence with a token, and the discriminator takes into input this sentence and has to try to detect the artefact token created by the generator. The goal of this training procedure is to reduce the combined loss of the two models, and once the training is completed, the discriminator model can be fine-tuned to solve a specific NLP task.

Finally, we decided to continue the MLM pretrain of the Italian base-cased version on domain-specific data. The company provided us with approximately 18,000 sentences corresponding to the subtitles of their health and workplace security courses. We continued the MLM pre-training task for 100 epochs with 20% of the dataset in validation. Then, we tried to go on with the pre-training on the same base models using a domain-specific vocabulary extension. Firstly, we analysed the fine-tuning dataset and found that 63% of the unique words were split into multiple tokens, and these words were often the most domain-specific and thus the ones that ideally contributed remarkably to the meaning of the sentence. Our idea was that splitting these words into subtokens was damaging performance; therefore, we added the words most frequently used in the fine-tuning dataset (with more than 100 repetitions). The 198 words added are mostly domain-specific words such as *isolanti* (insulators), *amianto* (asbestos), or *ustioni* (burns), and these new embeddings were pre-trained with the MLM task with the data described above.

In order to evaluate these two models, we analysed how the perplexity changes over the MLM task epochs; perplexity is a measurement of how well a probabilistic model predicts a sample and, therefore, the best value is 1, The higher the value of this metric, the less certain the model is of its prediction; in our case, the first model reached the best perplexity over the validation dataset, equal to 4.58 in 10 epochs, whereas the second model reached a value of 4.45 after 4 epochs.

As a classifier input we used the two BERT model outputs: the pooler output vector and the last hidden state matrix (except for the ELECTRA model, which returns only the second one). The first vector is obtained by manipulating the last [CLS] token embedding by the BertPooler layer. This layer is a simple feed-forward layer, with a hyperbolic tangent activation function, which does not modify the vector length, whereas the second output contains the embedding of each token computed by the last encoder. From these two outputs we tried five different inputs for the classification layer: the BertPooler output; the plain [CLS] vector; and the mean, minimum, and maximum values of the embeddings of the tokens. All the hyperparameter sets and the different training configurations which were considered in our experiments are reported in Appendix A.

7.2. Selection Results

In Figure 5, we show the results of the models described in the previous section. Given the high number of configurations, we aggregated them in five different groups for clarity's sake and, for each group, we visualized the best performing model. The groups are the following:

- the ITALIAN group, which is formed by four pre-trained models for Italian but without any further optimization;
- the MULTILINGUAL group, which contains two multilingual BERT models;
- the PRETRAINED group, which contains the two best models for the Italian language and onto which we performed the additional pre-training tasks, with (W-MLM model) and without (MLM model) the domain-specific words;
- the ELECTRA and ROBERTA models, including the homonymous pre-trained models.

As we can see on the left of Figure 5, the BERT models perform slightly better than ELECTRA and ROBERTA. In detail, the best Italian models reached a 0-F1 value over the

tasks equal to 0.688. We have approximately the same value for the PRETRAINED group, equal to 0.680. Therefore, the continued pre-training and the addition of domain-specific words do not improve the performance so much. Similar performances are achieved by multilingual models: in this case, the 0-F1 score is equal to 0.683, and this result is probably due to the fact that Italian is the 9th language, over 102 languages, in terms of frequency in the dataset used to train the multilingual models. Thus, if in the future the scoring system will be applied to more languages, the use of one of these models should not compromise the performance. For the last two groups, the performance is even worse: the ELECTRA model reaches a max 0-F1 of 0.623 and the ROBERTA model of 0.572, and this is probably due to the fact that these two models do not perform a pretraining for the [CLS] token, whereas BERT does with the next-sentence prediction task. Of these two models, ROBERTA obtains the worst performance. This is probably due to the fact that it is a multilingual model, while ELECTRA is specifically trained for the Italian language.

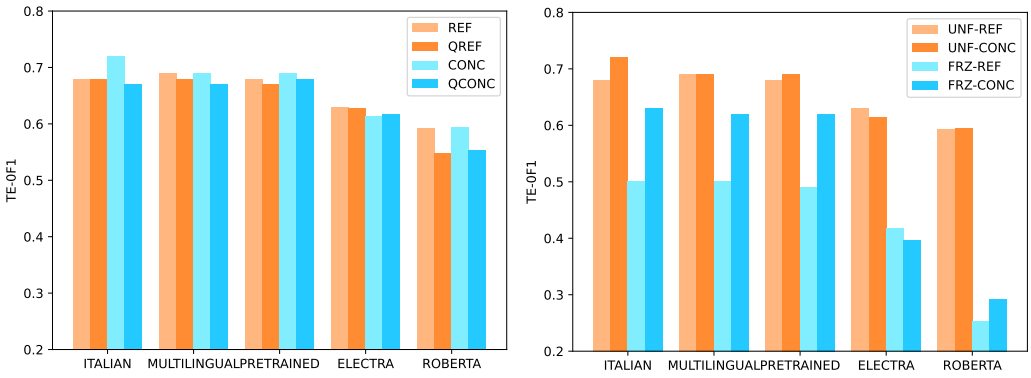

Figure 5. Performance analysis of the BERT models chosen as the basis for our artificial intelligence system. On the left, we compare the REF, QREF, CONC, and the QCONC fine-tuned models. On the right, we compare (for the REF and CONC models) frozen and fine-tuned versions of such models.

In terms of the different input configurations, we can see that there is no benefit in also adding the question to the model input. In fact, most of the QREF and QCONC models perform worse than the respective models without any questions. However, we claim that implementing a CONC model, into which the model has to identify the single concepts that compose the answer, is a good choice. In fact, these models perform similarly or even better than the REF models. Moreover, they are more explainable and the final user can see which concepts are identified by the model and which ones are not present. The best performing model, with 0.72 in terms of 0-F1, is the CONC model for the Italian XXL uncased BERT.

Another experiment we conducted is limiting the fine-tuning operations on BERT. In particular, we have *frozen* the BERT models, i.e., we only trained the final classification layer without updating all the BERT weights. In this way, each model has only to rely on the information encoded during the generic pre-training phase, without the possibility of learning from our datasets. This operation reduces the time and the computational cost of the fine-tuning procedure, but it also drastically impairs the overall performance. As can be seen on the right of Figure 5, all frozen models (in blue) obtain lower and insufficient performance in terms of 0-F1. In particular, the ROBERTA models suffer the most from freezing, with a reduction of 30 points in 0-F1, for both REF and CONC models. The Italian models lose approximately 18 points for the REF model and 9 points for the CONC model, and the same conclusions can be drawn for the other language models, where the models that leverage the concepts lose less performance with respect to the reference answer-based ones. In light of these results, we claim that updating the BERT parameters during the fine-tuning is necessary to achieve good performance. Please note that in Figure 5 we report only the performance of the REF and CONC versions of our models for clarity's sake. However, we have very similar results for the QCONC and QREF versions.

As concerns the classifier input, the models perform quite the same over all the values; the manipulation of the last hidden-state matrix gives better results than the pooler output or the plain CLS vector. The best performance, 0-F1 equal to 0.63, is achieved by applying the max pooling operation over the tokens embedding, followed by the application of the mean pooling operation that reduces the 0-F1 score by 1 point.

The balancing of the training dataset experiments worsens the performance by more than 2 points, from 0.636 to 0.613. This behaviour is probably due to the massive reduction of the training dataset, approximately 60% of the data; probably, with less strict balancing and less data removal, we could achieve better results. Another solution to this problem could be opposite operation, the data augmentation on the 0 labelled data; this operation is not trivial for textual data and is even more difficult in the semantic field. For the other hyperparameters, the dropout rate and the maximum tokenization length, no differences worthy of discussion have been found.

7.3. Best Models

As for the results of our grid search, we report the configuration and performance of the five best models in Table 3. In order to compare the results, we aggregate the concepts-based models by the user answers, multiplying each binary prediction (concept present, concept absent) by its relative weight, and summing them. If the sum is greater than the 80 threshold, we set the prediction value for the answer to 1, otherwise to 0.

Table 3. Best five model configurations that maximize the 0-F1 score on the test set. In this table, the base BERT model, the classifier input, and the hyperparameters values are reported. The best values of 0-F1 and 1-F1 are highlighted in bold.

Model	BERT	Classifier Input	Max. Length	Balanced	Dropout	Freeze	0-F1	1-F1
CONC	IT-XXL-UNC	Pooler	512	True	0.1	False	**0.72**	**0.87**
CONC	W-MLM	Min Emb.	128	True	0.1	False	0.69	**0.87**
CONC	ITA-UNC	Min Emb.	512	False	0.4	False	0.69	0.86
CONC	MULTI-UNC	Pooler	512	True	0.2	False	0.69	0.86
REF	MULTI-UNC	Min Emb.	512	True	0.2	False	0.69	0.85

As we can see in Table 3, the results are very similar. Three out of five of the best models leverage an Italian version of BERT; even if the manipulation of the embedding matrix with the max and mean operations is on average better than the other values, the best model takes advantage of the pooler output and the others prefer the minimum manipulation. Finally, even if on average the balancing of the training dataset worsens performance, four out of five best models are trained with the smaller and balanced dataset.

As we can see, the best CONC model reaches a 0.72 of 0-F1, whereas the best REF model stops at 0.69. The best QREF and QCONC models perform even less well; the first has a 0-F1 equal to 0.70 and the second equal to 0.68. In Table A2 in Appendix A, we report the complete sets of hyperparameters for the best CONC, QCONC, REF, and QREF models.

Besides the fact that the best model is concept-based, there are other reasons why the company choses to implement a concept-based model over the one based on the reference response. With concepts, human teachers can easily and quickly assess automatic corrections. From their point of view, it is easier to decide if a set of small key concepts is contained in the user's answers with respect to comparing two long and complex answers. The second main reason is a direct consequence of the classification task; with the REF and QREF models, the artificial intelligence system can tell human evaluators if an answer by a user has a score greater than or less than 80, but not the real score. With concept-based models, human correctors can assign a weight to the concepts and the user's answer score can be more accurate. For example, if a question has 4 concepts and each of them has a weight equal to 25, the automatic corrector can produce a score equal to 0, 25, 50, 75, or 100, depending on how many concepts the model recognises as present in the user's response.

8. Real-World Implementation

The Mega Italia Media e-learning platform, DynDevice, is a software written in PHP and JavaScript. With this platform, the users can access their courses through any web browser or mobile devices, and once they have finished the course, they can take the test. If a test is composed only of multiple-choice questions, the users receive the result in a few seconds; otherwise, if the test consists of open-ended questions, the correction process requires human intervention. Once the test is submitted, the human tutors receive an Email notification and have to manually evaluate the entire test. When they have finished scoring each question, the system sends the test corrected to the users, and if the final mark is at least over 60 out of 100, they have passed the test, and if not, they have to retry it.

With the introduction of the automatic corrector, the test evaluation procedure changed. As we can see in Figure 6, we introduced an intermediate software, called AI-Middleware, that handles the communication between DynDevice and deep learning models. So as to implement the middleware, we chose to leverage the Python library Flask, a lightweight WSGI (Web Server Gateway Interface) web application framework. Released in 2010, Flask has become over the past years one of the most popular frameworks to build web sites with Python.

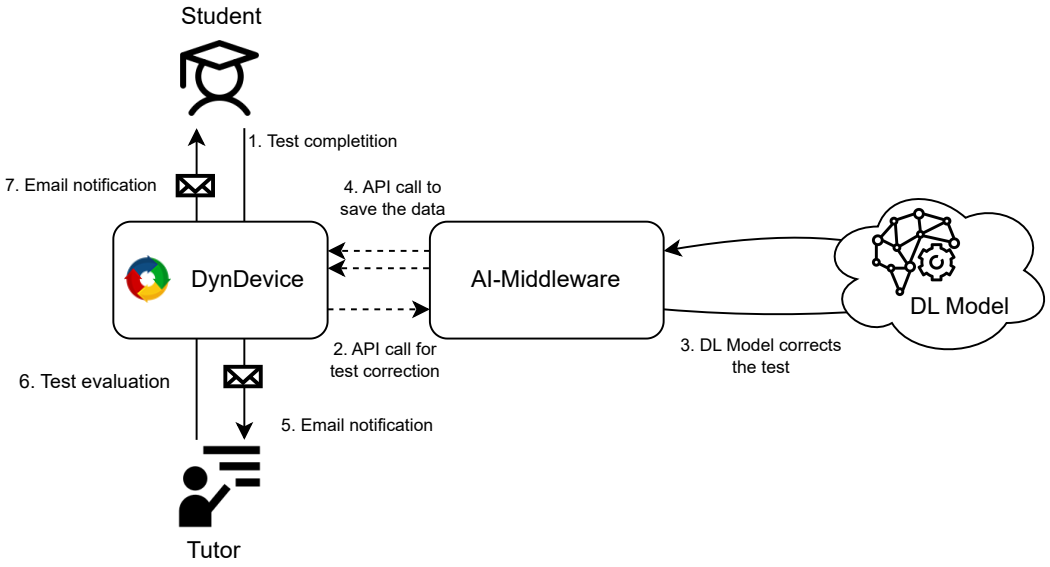

Figure 6. Architecture implemented for the test correction. It is made up of three main components: the platform DynDevice, the AI-Middleware, and the deep learning model. The arrows represent how they interact with the others and with the users (both the student and the tutor). Using straight arrows, we represent an interaction through a web platform or an Email, whereas by using dashed arrows we represent an API call. We also report the necessary numbered steps that the architecture follows to correct a test.

In the era of client–server architecture, APIs are often used to exchange information and services between two parties: the one who requires the service, the client, and the other who provides it, the server. While the server computes the answer, the client remains on hold; in our scenario, a test evaluation requires at least 15 s up to a minute. This number changes from test to test and depends on several factors, such as the weight of the JSON file exchanged or the internet connection speed. However, the main idle time is due to the number of predictions the deep learning model has to make (which depends on the combination of the user's responses and related concepts). Since DynDevice is a website, it is not possible for us to keep the client (the browser) on hold for more than a dozen seconds to avoid SEO penalties. Therefore, we implemented a dual API communication channel: first, DynDevice makes an API call to the middleware to send the test to correct, and once evaluated, the middleware makes an API call to DynDevice to save the results.

In our middleware, we implemented a small set of REST APIs that handle authentication operations with DynDevice and test correction. The scenario of a test correction is shown in Figure 6, and each step is enumerated:

1. at the end of the course, the user submits the open-ended questions-based test on the DynDevice platform;
2. the system makes an API call to the AI-Middleware to request the test correction. In an attachment to the call, it sends a JSON file containing the user answers and the relative concepts;
3. leveraging the deep learning model, the AI-Middleware provides the predictions: 1 if the concept is contained in the user's answer, 0 otherwise;
4. the results are sent back to DynDevice and saved in the company database through two API calls, one mandatory to authenticate to the DynDevice API service, and one to send the results;
5. the pre-evaluated test is sent to the human tutor through Email;
6. the tutor evaluates the test and confirms or changes the evaluation that the deep learning model made;
7. the user is notified of passing or failing the test by Email.

Before the introduction of the automatic corrector, Steps 2, 3, and 4 were skipped, and the human tutor was only notified of the presence of a new test to correct. Even if the number of steps increases with the introduction of the automatic corrector, the amount of time required to correct a single test is reduced by two-thirds.

Real-World Experimental Results

In the past 6 months, Mega Italia Media have used the artificial intelligence system to evaluate the open-ended questions-based test and, simultaneously, increased the number of questions with the concepts extracted. So as to evaluate the model predictions, we use as a label the human evaluation of the concept presence. In Table 4, we report precision, the percentage of class prediction correctly found, recall, the percentage of real class data correctly found, F1 score, the harmonic mean between the two previous metrics over the two classes, and global accuracy. As we can see from this column, the strong imbalance in the dataset is noteworthy: only 16% of the concepts are assessed as absent by the human evaluator, and this imbalance is even stronger than the one we have in the training and test sets.

Table 4. Evaluation of the best concepts-based models on the test submitted by the users in the last 6 months. The support column indicates the number of instances belonging to each class and (for the accuracy and the metrics average) the total number of test instances.

	Precision	**Recall**	**F1 Score**	**Support**
0	0.35	0.54	0.43	360
1	0.90	0.81	0.85	1851
accuracy			0.76	2211
macro avg	0.63	0.67	0.64	2211
weighed avg	0.81	0.76	0.78	2211

By analysing the results, we can see the model detects the concepts presence in users' answers well, given the F1 score over the 1 class equal to 0.85, but struggles to identify when a concept is absent. These results are consistent with those obtained in Table 3, even if the value of the F1 score on the 0 class is considerably lower. Globally, the model performs well, with 76% accuracy and the 0.78 value of the weighed F1 score, which means that the human evaluator had to change the prediction of the model only for 25% of the user responses.

Multiple reasons are behind this behaviour. Firstly, the data collected in the past months include (in addition to questions already seen by the model in training) new

questions, with the relative new concepts. These new concepts and questions have never been processed, and given that our model is fine-tuned over a few thousands of data, it is natural to think that it may not have acquired all the necessary knowledge about workplace health and safety, along with the fact that the addition of approximately 40% of new questions has worsened the performance. Probably with a larger number of answers and especially a dataset balanced between sufficient and insufficient answers, better results would have been obtained.

The second reason for this worsening may be found in the different labelling procedures. In fact, the original data used for the prototype were not directly labelled for each concept. Instead, to speed up the dataset creation process, these labels were derived from the overall score of the entire question; if the score was greater than 80 out of 100, we deduced that every concept required was present. If the score was lower than 80, concepts were considered absent. However, in this new real-world evaluation, we label the presence of the single concepts in the user's responses more carefully, evaluating each one of them. Therefore, these differences in the labelling process between fine-tuning and real-world evaluation can lead to a deterioration in performance. Overall, even with this low F1 score in the 0 class, the artificial intelligence system created meets the company's expectations. This is because we satisfy one of the main requirements, which is to correctly detect when a concept is included in the user's answer. This requirement was introduced because in most cases the users answered the questions correctly and passed the tests without too much trouble. Thus, with a precision of 0.90 on the 1 class, we correctly met this goal, and all this is reflected in the speed of test correction and the reduction of human workload.

In terms of time and user experience, the two domain experts who are responsible for the test correction claim that this system reduces the time of correction for humans by two thirds. According to the company, this can allow large savings in terms of time and money.

In addition, the introduction of this system has resulted in a significant reduction of the waiting time for users. In order to assess this reduction, we have examined the tests corrected by the human evaluators in the last two months (for a total of 106 tests and a total of 338 questions). During this period, the average waiting time for a test to be corrected without our system was approximately 12 working hours. On the contrary, when human evaluators are assisted by our automated system, the average waiting time is approximately 4 h. Although we cannot verify the correction time specifically, we claim that this reduction in the overall waiting time is a significant result. We expect that the application of our automatic system to a larger number of tests will make this gain in terms of time even more remarkable.

We have also performed a statistical comparison among our models based on Friedman's test [43]. The results presented on the top of Figure 7 show a statistically different performance of the CONC model (with a p-value lower than 0.01) with respect to QCONC, QREF, and REF models. On the other hand, the differences among these models are not statistically different in a significant way (the p-value is higher than 0.05). On the bottom, we show a simple diagram into which models are ordered from left to right in terms of 0-F1, and the best is on the left. This means that our choice of CONC as the best performing model is valid from a statistical point of view.

The last evaluation we conducted is measuring the performance of our system on the real-world dataset but divided into single questions. Given the high number of different questions, in Table 5 we provide a selection of six different examples of these questions: three which were already present in the training set of our application (on the top) and three which were not (on the bottom). Interestingly, the model performs well, also on questions which were never seen in training. This is probably due to the similarity of topics covered in different questions, given that they often belong to the same test or course. However, an interesting aspect is that the most generic questions ("What is the definition of electrical work?" and "What PPE is required to perform electrical work under voltage (low voltage)?") obtains a lower accuracy. This is probably due to the fact that the user can express an answer in many different ways, causing some issues to the automatic corrector.

From a more general and quantitative point of view, the system proved to be robust, with at least 25% of questions having 100% accuracy and 50% of questions exceeding 84%.

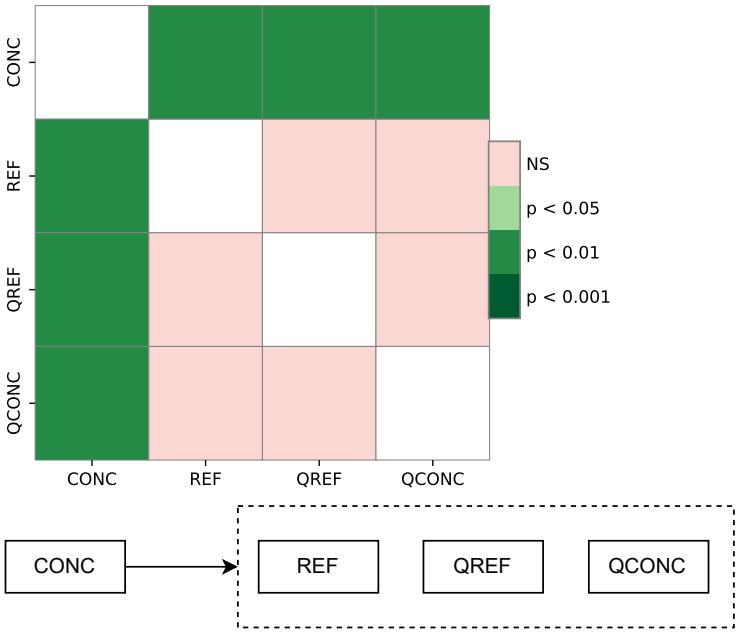

Figure 7. Pairwise analysis of Friedman's tests across the best models for CONC, QCONC, REF, and QREF configurations. The CONC model performs statistically differently from the REF, QREF, and QCONC models, with a *p*-value lower than 0.01. NS stands for no significant difference between the models in terms of performance. On the bottom, graphical representation of the Friedman test. The arrows show models with statistically different performance.

Table 5. Single question evaluation for six samples taken from the real-world dataset. The first three questions were also present in the training dataset, whereas the three in the second group were never seen by the 'model'.

Question	Num. Answers	Accuracy
What is the definition of electrical work?	116	0.66
What is the correct sequence of actions to take in first aid to an electrocution victim? What precautions must the rescuer take for himself?	108	0.75
When is an electrical installation considered low-voltage according to IEC 64-8? Can it cause injury to the human body?	68	0.91
Explain the differences between 'work under voltage', 'work above voltage' and 'work in close proximity' as defined in IEC 11-27 in relation to the performance of electrical work.	330	0.91
What is a work plan (in the context of electrical work as defined in IEC 11-27)? Can it be compared to the intervention plan?	243	0.79
What PPE is required to perform electrical work under voltage (low voltage)?	54	0.59

From a legal and ethical perspective, please note that all the answers provided by our system have to be validated by human teachers by law. Our original data cannot be shared due to privacy reasons.

9. Conclusions and Future Work

The e-learning sector has grown enormously in the last few years, and therefore having artificial intelligence tools to support employees has become necessary. Our work started from the study of the state-of-the-art about scoring systems, the field of natural language processing, and a deep study of the company workflow and user case. After that we gathered all the data available in the company's database and started to build the architecture of our model to train with a supervised approach. We defined four different

types of model, two leveraging the reference answers and two the concepts. After an extensive analysis and several experiments required to find the best combination of input, model, and hyperparameters, the results obtained are satisfactory but not optimal. The best model reached a score value of F1 on the test set of 0.78, with a higher value, approximately 0.86, on class 1, which indicates a response with a score above 80 out of 100.

In addition to building the best possible model, another important step was to integrate the possible system within the enterprise software and in the human tutor workflow. In order to do so, we decided to leverage on a client-server architecture that exchanges information through an API call. This choice was mainly due to high fault tolerance, high parallelisation, and easy scalability. Multiple API calls can be executed, so multiple tests can be corrected, and if an API call crashes, the other calls are not affected. Furthermore, in the future, it will be easy to extend the middleware with new features or add new API calls or different deep learning models to correct the tests.

Even considering the not optimal results on the test ground and the slightly worse performance in the real-world setting, we fulfilled the company's main requirements. The implemented automatic scoring system has significantly lightened the workload of the human tutor, reducing the time to correct a single test by two-thirds. This was due to the architectural choices made to increase the precision of the concept-present class. So, if a concept is present, as in the majority of cases, the human tutor has to briefly revise the correction without too much concern.

The artificial intelligence system created and integrated can obviously be further improved. As previously mentioned, the performance of the deep learning model can be improved with a deep fine-tuning with more data or by using a new architecture or techniques that take more into account the semantics of a sentence, such as SBERT [24] or Semantic-Aware BERT [44]. In addition to these already developed technologies, the semantic similarity field is an active and open research field, and much can be done to solve this not trivial task.

Another possible new future work could be the introduction of explainability, i.e., identify where in the user's response the automatic scoring system has detected the presence of the concept, similarly to what has been done in [45–47]. This new feature will even increase the time that human tutors invest in correcting tests and, at a glance, these people could validate the prediction given by the classifier. Moreover, a more intuitive explanation of the answers by our system could prevent unwanted or discriminatory behaviour [48–50].

Finally, new architectures and pre-trained language models based on GPT [51] have been introduced in the last few years. Since they obtained promising results in question answering tasks, we aim to investigate the use of these models in our context.

Author Contributions: Conceptualization, A.E.G., L.P. and I.S.; Software, N.A.; Validation, M.O.; Investigation, N.A. and L.P.; Resources, N.A.; Data curation, L.S.; Writing—original draft, N.A. and L.P.; Writing—review & editing, I.S. All authors have contributed equally to the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the MIUR "Fondo Departments of Excellence 2018–2022" of the DII Department at the University of Brescia, Italy, EU H2020 project AIPlan4EU (GA n. 101016442) and EU ICT-48 2020 project TAILOR (No. 952215). The *IBM Power Systems Academic Initiative* substantially contributed to the experimental analysis.

Data Availability Statement: The source code of this work is available at the following GitHub repository: https://github.com/nicolarici/ASAG-IT-WS (accessed on 8 August 2023). The data used for training and testing our system could not be made available (except for some selected sampled) for privacy reasons.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Deep Learning Models Hyperparameters

Table A1 shows the hyperparameter space searched with the grid search experimentation. For clarity, the tables show the abbreviations of the BERT model names. With 'IT' we

denoted the BERT models in the Italian language; with 'UNC' the uncased models; with 'CAS' the cased models; with 'MULTI' the multilingual models; with 'MLM' the model with extended pretrain; and with 'WMLM' the model with extended pretrain and the introduction of domain-specific words. Thus, the "IT-XXL-UNC" model is the Italian XXL uncased version of BERT, and so on. As the maximum tokenization length, we tried two values, 128, because the mean of the dataset tokenization distribution is equal to 126, and 512, the maximum value accepted by BERT, because the 99th percentile of the tokenization distribution is equal to 416. This parameter controls the length of the token list in input to the BERT model; the sequences longer than this value are truncated, whereas the shorter ones are padded. As a dropout rate for the dropout layer of the classifier, we tested the three values shown in Table A1. Another experiment was trying to reduce the imbalance on the training set through a random subsampling operation on the 1 class labelled data; in order to balance the dataset, we removed approximately 60% of the data for the 1 class. Finally, to speed up the tuning process and reduce the computational cost, we tried to freeze the BERT weights, i.e., we only trained the final classification layer without updating the BERT weights.

Table A1. Grid search hyperparameter space.

Hyperparameter	Value
BERT Model	IT-UNC, IT-CAS, IT-XXL-UNC, IT-XXL-CAS, MULTI-UNC, MULT-CAS, ROBERTA, ELECTRA, MLM, W-MLM
Classifier Input	Pooler, CLS, Emb. Mean, Emb. Min, Emb. Max
Maximum Tokenization length	128, 512
Dropout Rate	0.1, 0.2, 0.4
Training set balanced	True, False
Freeze BERT weight	True, False

In Table A2, we report the best set of hyperparameters for each model presented in Section 6.

Table A2. Set of hyperparameters for the best models we obtained for each configuration.

Model	BERT	Classifier Input	Max. Length	Balanced	Dropout	Freeze
REF	MULTI-UNC	Min Emb.	512	True	0.2	False
QREF	MULTI-UNC	Min Emb.	128	False	0.1	False
CONC	IT-XXL-UNC	Pooler	512	True	0.1	False
QCONC	W-MLM	Min Emb.	512	False	0.1	False

References

1. Arici, N.; Gerevini, A.E.; Putelli, L.; Serina, I.; Sigalini, L. A BERT-Based Scoring System for Workplace Safety Courses in Italian. In *Lecture Notes in Computer Science, Proceedings of the AIxIA 2022—Advances in Artificial Intelligence—XXIst International Conference of the Italian Association for Artificial Intelligence, AIxIA 2022, Udine, Italy, 28 November–2 December 2022*; Dovier, A., Montanari, A., Orlandini, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2022; Volume 13796, pp. 457–471. [CrossRef]
2. Devlin, J.; Chang, M.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019; Burstein, J., Doran, C., Solorio, T., Eds.; Long and Short Papers; Association for Computational Linguistics: Cambridge, MA, USA, 2019; Volume 1, pp. 4171–4186.
3. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All you Need. In Proceedings of the Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, 4–9 December 2017; Guyon, I., von Luxburg, U., Bengio, S., Wallach, H.M., Fergus, R., Vishwanathan, S.V.N., Garnett, R., Eds.; Curran Associates Inc.: Red Hook, NY, USA, 2017; pp. 5998–6008.

4. Wang, Y.; Wang, C.; Li, R.; Lin, H. On the Use of Bert for Automated Essay Scoring: Joint Learning of Multi-Scale Essay Representation. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, USA, 10–15 July 2022; Carpuat, M., de Marneffe, M., Ruíz, I.V.M., Eds.; Association for Computational Linguistics: Cambridge, MA, USA, 2022; pp. 3416–3425. [CrossRef]
5. Beltagy, I.; Lo, K.; Cohan, A. SciBERT: A Pretrained Language Model for Scientific Text. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 November 2019; Inui, K., Jiang, J., Ng, V., Wan, X., Eds.; Association for Computational Linguistics: Cambridge, MA, USA, 2019; pp. 3613–3618. [CrossRef]
6. Xie, Q.; Dai, Z.; Hovy, E.H.; Luong, T.; Le, Q. Unsupervised Data Augmentation for Consistency Training. In Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Virtual Event, 6–12 December 2020; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., Eds.; Curran Associates Inc.: Red Hook, NY, USA, 2020.
7. Putelli, L.; Gerevini, A.E.; Lavelli, A.; Olivato, M.; Serina, I. Deep Learning for Classification of Radiology Reports with a Hierarchical Schema. In *Procedia Computer Science, Proceedings of the 24th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems KES-2020, Virtual Event, 16–18 September 2020*; Cristani, M., Toro, C., Zanni-Merk, C., Howlett, R.J., Jain, L.C., Eds.; Elsevier: Amsterdam, The Netherlands, 2020; Volume 176, pp. 349–359. [CrossRef]
8. Calvo, R.A.; Milne, D.N.; Hussain, M.S.; Christensen, H. Natural language processing in mental health applications using non-clinical texts. *Nat. Lang. Eng.* **2017**, *23*, 649–685. [CrossRef]
9. Ramaswamy, S.; DeClerck, N. Customer Perception Analysis Using Deep Learning and NLP. *Procedia Comput. Sci.* **2018**, *140*, 170–178. [CrossRef]
10. Oh, Y.K.; Yi, J. Asymmetric effect of feature level sentiment on product rating: An application of bigram natural language processing (NLP) analysis. *Internet Res.* **2022**, *32*, 1023–1040. [CrossRef]
11. Zubani, M.; Sigalini, L.; Serina, I.; Putelli, L.; Gerevini, A.E.; Chiari, M. A Performance Comparison of Different Cloud-Based Natural Language Understanding Services for an Italian e-Learning Platform. *Future Internet* **2022**, *14*, 62. [CrossRef]
12. Reddy Karri, S.P.; Santhosh Kumar, B. Deep Learning Techniques for Implementation of Chatbots. In Proceedings of the 2020 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 22–24 January 2020. [CrossRef]
13. Liu, X.; Eshghi, A.; Swietojanski, P.; Rieser, V. Benchmarking Natural Language Understanding Services for Building Conversational Agents. In *Lecture Notes in Electrical Engineering, Proceedings of the Increasing Naturalness and Flexibility in Spoken Dialogue Interaction—10th International Workshop on Spoken Dialogue Systems, IWSDS 2019, Syracuse, Italy, 24–26 April 2019*; Marchi, E., Siniscalchi, S.M., Cumani, S., Salerno, V.M., Li, H., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 714, pp. 165–183. [CrossRef]
14. Ke, Z.; Ng, V. Automated Essay Scoring: A Survey of the State of the Art. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, 10–16 August 2019; Kraus, S., Ed.; International Joint Conferences on Artificial Intelligence: Darmstadt, Germany, 2019; pp. 6300–6308. [CrossRef]
15. Haller, S.; Aldea, A.; Seifert, C.; Strisciuglio, N. Survey on Automated Short Answer Grading with Deep Learning: From Word Embeddings to Transformers. *arXiv* **2022**, arXiv:2204.03503.
16. Mohler, M.; Bunescu, R.C.; Mihalcea, R. Learning to Grade Short Answer Questions using Semantic Similarity Measures and Dependency Graph Alignments. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; Lin, D., Matsumoto, Y., Mihalcea, R., Eds.; The Association for Computational Linguistics: Cambridge, MA, USA, 2011; pp. 752–762.
17. Pribadi, F.S.; Adji, T.B.; Permanasari, A.E.; Mulwinda, A.; Utomo, A.B. Automatic short answer scoring using words overlapping methods. *AIP Conf. Proc.* **2017**, *1818*, 020042.
18. Gomaa, W.; Fahmy, A. Short Answer Grading Using String Similarity And Corpus-Based Similarity. *Int. J. Adv. Comput. Sci. Appl.* **2012**, *3*, 11. [CrossRef]
19. Suzen, N.; Gorban, A.N.; Levesley, J.; Mirkes, E.M. Automatic Short Answer Grading and Feedback Using Text Mining Methods. *arXiv* **2018**, arXiv:1807.10543. Available online: http://xxx.lanl.gov/abs/1807.10543 (accessed on 8 August 2023).
20. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. In Proceedings of the 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, AZ, USA, 2–4 May 2013; Bengio, Y., LeCun, Y., Eds.; Workshop Track Proceedings; 2013.
21. Jin, P.; Zhang, Y.; Chen, X.; Xia, Y. Bag-of-embeddings for text classification. In Proceedings of the IJCAI, New York, NY, USA, 9–15 July 2016; Volume 16, pp. 2824–2830.
22. Rudkowsky, E.; Haselmayer, M.; Wastian, M.; Jenny, M.; Emrich, Š.; Sedlmair, M. More than bags of words: Sentiment analysis with word embeddings. *Commun. Methods Meas.* **2018**, *12*, 140–157. [CrossRef]
23. Galke, L.; Scherp, A. Bag-of-words vs. graph vs. sequence in text classification: Questioning the necessity of text-graphs and the surprising strength of a wide MLP. *arXiv* **2021**, arXiv:2109.03777.

24. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 November 2019; Inui, K., Jiang, J., Ng, V., Wan, X., Eds.; Association for Computational Linguistics: Cambridge, MA, USA, 2019; pp. 3980–3990.
25. Bao, W.; Bao, W.; Du, J.; Yang, Y.; Zhao, X. Attentive Siamese LSTM Network for Semantic Textual Similarity Measure. In Proceedings of the 2018 International Conference on Asian Language Processing (IALP), Bandung, Indonesia, 15–18 November 2018; pp. 312–317. [CrossRef]
26. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, Doha, Qatar, 25–29 October 2014; A meeting of SIGDAT, a Special Interest Group of the ACL; Moschitti, A., Pang, B., Daelemans, W., Eds.; Association for Computational Linguistics: Cambridge, MA, USA, 2014; pp. 1532–1543. [CrossRef]
27. Hassan, S.; Fahmy, A.A.; El-Ramly, M. Automatic Short Answer Scoring based on Paragraph Embeddings. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 10. [CrossRef]
28. Gomaa, W.H.; Fahmy, A.A. Ans2vec: A Scoring System for Short Answers. In *Advances in Intelligent Systems and Computing, Proceedings of the International Conference on Advanced Machine Learning Technologies and Applications, AMLTA 2019, Cairo, Egypt, 28–30 March 2019*; Hassanien, A.E., Azar, A.T., Gaber, T., Bhatnagar, R., Tolba, M.F., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 921, pp. 586–595. [CrossRef]
29. Kiros, R.; Zhu, Y.; Salakhutdinov, R.; Zemel, R.S.; Urtasun, R.; Torralba, A.; Fidler, S. Skip-Thought Vectors. In Proceedings of the Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montreal, QC, Canada, 7–12 December 2015; Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R., Eds.; MIT Press: Cambridge, MA, USA, 2015, pp. 3294–3302.
30. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
31. Prabhudesai, A.; Duong, T.N.B. Automatic Short Answer Grading using Siamese Bidirectional LSTM Based Regression. In Proceedings of the IEEE International Conference on Engineering, Technology and Education, TALE 2019, Yogyakarta, Indonesia, 10–13 December 2019; IEEE: New York, NY, USA, 2019; pp. 1–6.
32. Kumar, S.; Chakrabarti, S.; Roy, S. Earth Mover's Distance Pooling over Siamese LSTMs for Automatic Short Answer Grading. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, VIC, Australia, 19–25 August 2017; Sierra, C., Ed.; International Joint Conferences on Artificial Intelligence: Darmstadt, Germany, 2017; pp. 2046–2052. [CrossRef]
33. Sung, C.; Dhamecha, T.I.; Saha, S.; Ma, T.; Reddy, V.; Arora, R. Pre-Training BERT on Domain Resources for Short Answer Grading. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 November 2019; Inui, K., Jiang, J., Ng, V., Wan, X., Eds.; Association for Computational Linguistics: Cambridge, MA, USA, 2019; pp. 6070–6074.
34. Lun, J.; Zhu, J.; Tang, Y.; Yang, M. Multiple Data Augmentation Strategies for Improving Performance on Automatic Short Answer Scoring. In Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, 7–12 February 2020; AAAI Press: Washington, DC, USA, 2020; pp. 13389–13396.
35. Del Gobbo, E.; Guarino, A.; Cafarelli, B.; Grilli, L. GradeAid: A framework for automatic short answers grading in educational contexts—Design, implementation and evaluation. *Knowl. Inf. Syst.* **2023**, 1–40. [CrossRef] [PubMed]
36. Brika, S.K.M.; Chergui, K.; Algamdi, A.; Musa, A.A.; Zouaghi, R. E-learning research trends in higher education in light of COVID-19: A bibliometric analysis. *Front. Psychol.* **2022**, *12*, 762819. [CrossRef] [PubMed]
37. Mouratidis, K.; Papagiannakis, A. COVID-19, internet, and mobility: The rise of telework, telehealth, e-learning, and e-shopping. *Sustain. Cities Soc.* **2021**, *74*, 103182. [CrossRef] [PubMed]
38. Edem Adzovie, D.; Jibril, A.B. Assessment of the effects of Covid-19 pandemic on the prospects of e-learning in higher learning institutions: The mediating role of academic innovativeness and technological growth. *Cogent Educ.* **2022**, *9*, 2041222. [CrossRef]
39. Ethayarajh, K. How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 November 2019; Inui, K., Jiang, J., Ng, V., Wan, X., Eds.; Association for Computational Linguistics: Cambridge, MA, USA, 2019; pp. 55–65.
40. Peters, M.E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep Contextualized Word Representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, LA, USA, 1–6 June 2018; Walker, M.A., Ji, H., Stent, A., Eds.; Long Papers; Association for Computational Linguistics: Cambridge, MA, USA, 2018; Volume 1, pp. 2227–2237.
41. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692. Available online: http://xxx.lanl.gov/abs/1907.11692 (accessed on 8 August 2023).

42. Clark, K.; Luong, M.; Le, Q.V.; Manning, C.D. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In Proceedings of the 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, 26–30 April 2020. Available online: OpenReview.net (accessed on 8 August 2023).
43. Sheldon, M.R.; Fillyaw, M.J.; Thompson, W.D. The use and interpretation of the Friedman test in the analysis of ordinal-scale data in repeated measures designs. *Physiother. Res. Int.* **1996**, *1*, 221–228. [CrossRef] [PubMed]
44. Zhang, Z.; Wu, Y.; Zhao, H.; Li, Z.; Zhang, S.; Zhou, X.; Zhou, X. Semantics-Aware BERT for Language Understanding. In Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, 7–12 February 2020. AAAI Press: Washington, DC, USA, 2020; pp. 9628–9635.
45. Putelli, L.; Gerevini, A.E.; Lavelli, A.; Maroldi, R.; Serina, I. Attention-Based Explanation in a Deep Learning Model For Classifying Radiology Reports. In *Lecture Notes in Computer Science, Proceedings of the Artificial Intelligence in Medicine—19th International Conference on Artificial Intelligence in Medicine, AIME 2021, Virtual Event, 15–18 June 2021*; Tucker, A., Abreu, P.H., Cardoso, J.S., Rodrigues, P.P., Riaño, D., Eds.; Springer: Berlin/Heidelberg, Germany, 2021; Volume 12721, pp. 367–372. ._42. [CrossRef]
46. Putelli, L.; Gerevini, A.E.; Lavelli, A.; Mehmood, T.; Serina, I. On the Behaviour of BERT's Attention for the Classification of Medical Reports. In Proceedings of the 3rd Italian Workshop on Explainable Artificial Intelligence Co-Located with 21th International Conference of the Italian Association for Artificial Intelligence(AIxIA 2022), Udine, Italy, 28 November–3 December 2022; Musto, C., Guidotti, R., Monreale, A., Semeraro, G., Eds.; CEUR Workshop Proceedings; CEUR-WS: Aachen, Germany, 2022; Volume 3277, pp. 16–30.
47. Serina, L.; Putelli, L.; Gerevini, A.E.; Serina, I. Synonyms, Antonyms and Factual Knowledge in BERT Heads. *Future Internet* **2023**, *15*, 230. [CrossRef]
48. Hovy, D.; Prabhumoye, S. Five sources of bias in natural language processing. *Lang. Linguist. Compass* **2021**, *15*, e12432. [CrossRef] [PubMed]
49. Dusi, M.; Arici, N.; Gerevini, A.E.; Putelli, L.; Serina, I. Graphical Identification of Gender Bias in BERT with a Weakly Supervised Approach. In Proceedings of the Sixth Workshop on Natural Language for Artificial Intelligence (NL4AI 2022) Co-Located with 21th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2022), Udine, Italy, 30 November 2022; Nozza, D., Passaro, L.C., Polignano, M., Eds.; CEUR Workshop Proceedings; CEUR-WS: Aachen, Germany, 2022; Volume 3287, pp. 164–176.
50. Nadeem, M.; Bethke, A.; Reddy, S. StereoSet: Measuring stereotypical bias in pretrained language models. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, Virtual Event, 1–6 August 2021; Zong, C., Xia, F., Li, W., Navigli, R., Eds.; Long Papers; Association for Computational Linguistics: Cambridge, MA, USA, 2021; Volume 1, pp. 5356–5371.
51. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving language understanding by generative pre-training. Available online: https://openai.com/research/language-unsupervised (accessed on 8 August 2023).