Democratic and Popular Algerian Republic Ministry of Higher Education
and Scientific Research

YAHYA FARES UNIVERSITY OF MEDEA
FACULTY OF SCIENCES
Department of Mathematics and Computer Science



Final Master Project
Branch: Computer Science
Speciality: System Engineering and Web Technologies

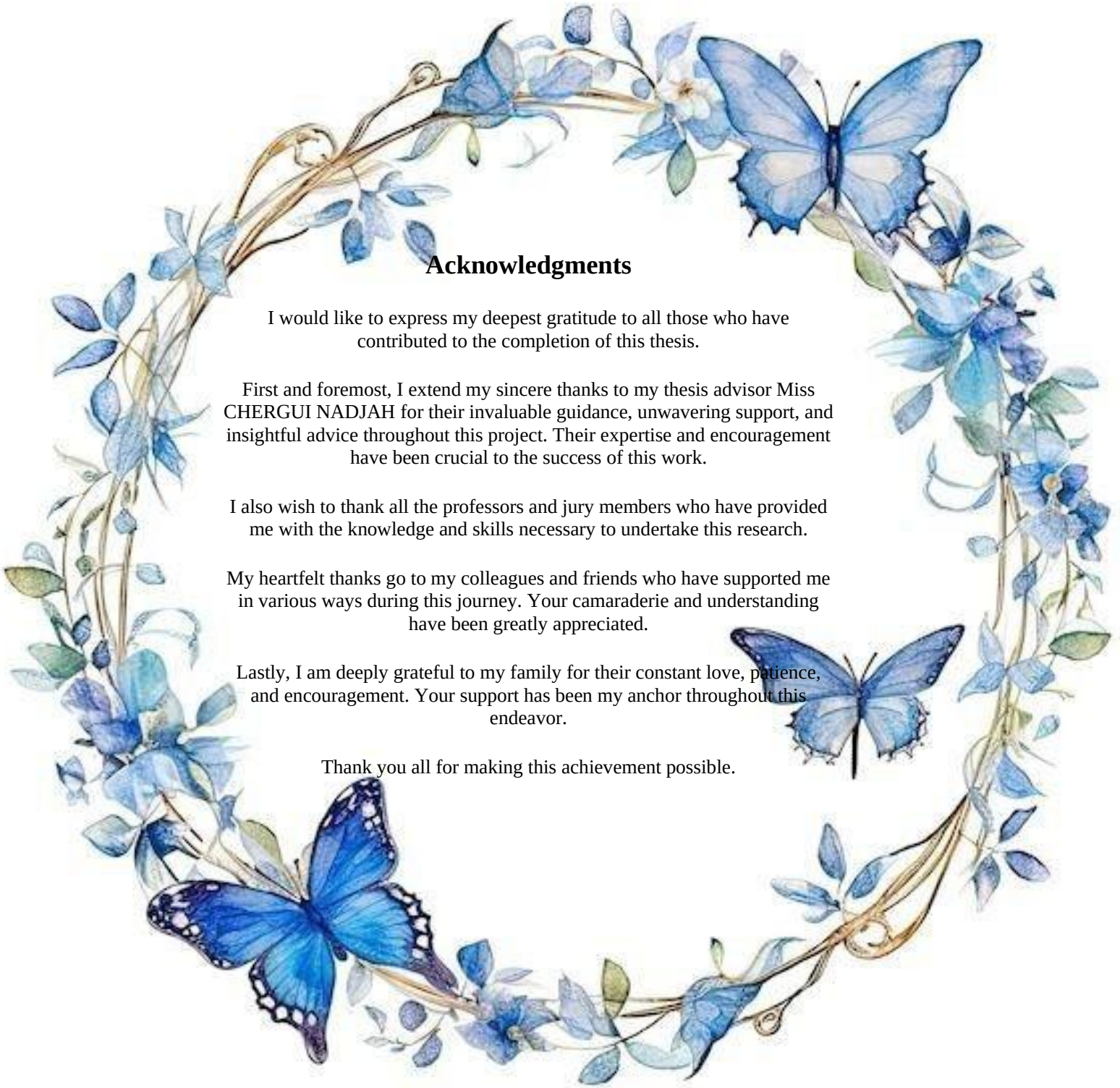
Text Comprehension for E-Learning Applications

Prepared and presented by :
BOUHARAOUA AMINA

supervisor :
CHERGUI NADJAH

Jury president : Mr. Abdoun Nabil
Reviewer: Mr. Benaissa Azzedine

university year
2023/2024



Acknowledgments

I would like to express my deepest gratitude to all those who have contributed to the completion of this thesis.

First and foremost, I extend my sincere thanks to my thesis advisor Miss CHERGUI NADJAH for their invaluable guidance, unwavering support, and insightful advice throughout this project. Their expertise and encouragement have been crucial to the success of this work.

I also wish to thank all the professors and jury members who have provided me with the knowledge and skills necessary to undertake this research.

My heartfelt thanks go to my colleagues and friends who have supported me in various ways during this journey. Your camaraderie and understanding have been greatly appreciated.

Lastly, I am deeply grateful to my family for their constant love, patience, and encouragement. Your support has been my anchor throughout this endeavor.

Thank you all for making this achievement possible.

بسم الله الرحمان الرحيم

اليوم مسك الختام و عبرة النهايات ، الحمد لله الذي ما نجحنا و ما علونا و ما توفقنا الا برضاه هو الملاذ و الملجأ ان ضاقت بنا الايام

بعد مسيرة دامت سنوات حملت في طياهتا الكثير من الصعوبات و المشقات و التعب

ها انا اليوم اقف عند عتبة تخرجي أقطف ثمار تعبي و ارفع قبعتي بكل فخر

اهدي هذا النجاح الى الذي زين اسمي باجل الالقاب، من دعمني بلا حدود و اعطاني بلا مقابل ، الى من علمني ان الدنيا كفاح و سلاحها العلم ، اعمي في مسيرتي و ملاذي بعد الله، لطالما عاهدته بهذا النجاح و ها انا اتممت وعدي الغالي أبي

الى من جعل الله الجنة تحت اقدامها و احتضنني قلبها قبل يديها و سهلت لي الشدائد بدعائها ، الى من بكيت على كتفها سر قوتي و نجاحي أمي

الى الداعمين الساندين ارضي صلبو و جداري المتين ، الى من قال فيهم الله" و اشدد عضدك بأخيك" اخي عبد الاله و خواتي خلود ، اسراء ، تسنيم

الى كل فرد من عائلتي من خالاتي و عماتي ، اعمام و اخوالي و كل ابنائهم .

ولا انسى رفقاتي التي شاركن معي هته الطريق ، الى من هون تعب السنين ها قد وصلنا اليوم للسبيل الذي يفرقنا و ها نحن بعد 5 سنوات نودع بعضنا و نمسح دموع الفرح و الحنين صديقاتي مريا ، فاطمة الزهراء ،أمينة، قطر الندى ، مفيدة، خولة ،مريم و يسرى.

الى كل من كان لي سندا من قريب او بعيد

ها انا اليوم انمم اول ثمارات نجاحي راجية من الله تعالى ان ينفعني بما علمني و ان يعلمني ما اجهله و يجعله حجة لي لا علي.

Abstract

Automatic Short Answers Grading system for MOOCs, focusing on the English language. This study leverages deep learning techniques to create a grading system that not only evaluates students' answers based on correctness but also comprehends the depth of their understanding and the contextual relevance of their responses. The methodology employed involves training various neural network models to analyze text responses, comparing their performance, and integrating the most effective model into the MOOC platform. The proposed system aims to enhance the accuracy of automated grading and reduce the workload of human graders. The effectiveness of the ASAG system is validated through a series of experiments, demonstrating its potential to support educators and improve the educational experience in MOOCs.

Key word:

MOOCs, Deep learning, Transfer learning, BERT, Elmo, Bi-LSTM.

Résume

Ce travail étudie le développement et la validation d'un système de notation automatique de réponses courtes (ASAG) pour les MOOCs, en se concentrant sur la langue anglaise. Avec l'adoption croissante des plateformes d'éducation en ligne, le besoin de méthodes d'évaluation efficaces et évolutives est primordial. Cette étude tire parti des techniques d'apprentissage profond pour créer un système de notation qui évalue non seulement les réponses des étudiants sur la base de leur exactitude, mais comprend également la profondeur de leur compréhension et la pertinence contextuelle de leurs réponses. La méthodologie employée implique la formation de différents modèles de réseaux neuronaux pour analyser les réponses textuelles, comparer leur performance et intégrer le modèle le plus efficace dans la plateforme MOOC. Le système proposé vise à améliorer la précision de la notation automatique et à réduire la charge de travail des correcteurs humains. L'efficacité du système ASAG est validée à travers une série d'expériences, démontrant son potentiel pour soutenir les éducateurs et améliorer l'expérience éducative dans les MOOCs.

Les mots clés:

MOOCs, L'apprentissage en profondeur, Transférer l'apprentissage ,BERT ,Elmo ,Bi-LSTM.

ملخص

يستعرض هذا التقرير تطوير وتحقيق نظام تصحيح تلقائي للإجابات القصيرة(ASAG)لدورات التعلم المفتوح عبر الإنترنت(MOOCs) ، مع التركيز على اللغةالإنجليزية. مع تزايد اعتماد منصات التعليم عبر الإنترنت، أصبح من الضروريوجود طرق تقييم فعالة وقابلة للتوسع. تستفيد هذه الدراسة من تقنيات التعلمالعميق لإنشاء نظام تقييم يقيم إجابات الطلاب ليس فقط بناءً على صحتها،ولكن أيضًا على فهمهم العميق ومدى ملاءمة إجاباتهم من حيث السياق. تشملمنهجية العمل تدريب نماذج متعددة من الشبكات العصبية لتحليل النصوص،ومقارنة أدائها، ودمج النموذج الأكثر كفاءة في منصةMOOCs .

يهدف النظام المقترح إلى تعزيز دقة التقييم الآلي وتقليل عبء العمل على المصححين البشريين. تم التحقق من فعالية نظامASAGمن خلال سلسلة من التجارب، ممايثبت إمكاناته في دعم المعلمين وتحسين تجربة التعليم فيMOOCs.

الكلمة الرئيسية:

MOOCs، التعلم العميق، نقل التعلم، BERT، Elmo، Bi-LSTM.

Acronyms

ACC: Accuracy

AI: Artificial Intelligence

BERT: Bidirectional Encoder Representations from Transformers

bi-LSTM: Bidirectional Long Short-Term Memory

Elmo: Embeddings from Language Models

EMD: Earth Mover's Distance

MAE: Mean Absolute Error

ML: Machine Learning

MOOCs: Massive Open Online Course

MRC: Machine Reading Comprehension

MSE: Mean Squared Error

NLP: Natural Language Processing

OER: Open Educational Resources

ONNX: Open Neural Network Exchange

PCC: Pearson Correlation Coefficient

RMSE: Root Mean Squared Error

Contents

Abstract **3**

Abstract **4**

1 Background and Literature Review **4**
1.1 Moocs definition . . . 4
1.1.1 MOOCs forms . . . 5
1.1.2 MOOCs Advantages . . . 5
1.2 MOOCs structure . . . 6
1.3 Assessment . . . 6
1.3.1 Phases of Assessment . . . 6
1.3.2 Types of Assessment in Education . . . 7
1.3.3 Automated Grading Systems . . . 7
1.3.4 Approaches to Automated Short Answer Grading (ASAG) Systems 8
1.4 Text Comprehension . . . 8
1.4.1 Text comprehension using deep learning . . . 8
1.5 Text Similarity Measurement . . . 8
1.5.1 Lexical Similarity . . . 9
1.5.2 Semantic Similarity Measures . . . 10
1.5.3 Advanced Embedding-Based Measures . . . 10
1.5.4 Hybrid similarity . . . 11
1.6 Regression . . . 11
1.6.1 Regression Models . . . 11
1.7 Embedding . . . 11
1.8 Artificial intelligence . . . 12
1.9 Machine learning . . . 13
1.9.1 Significance of Machine Learning . . . 13
1.9.2 Type of Machine Learning . . . 14
1.10 Deep learning . . . 15
1.10.1 Benefits of Deep learning . . . 15
1.10.2 DEEP MODELS in text comprehension description . . . 16
1.11 Transfer learning . . . 20
1.11.1 What Is Transfer Learning? . . . 20
1.11.2 Categories . . . 21
1.11.3 Why Use Transfer Learning . . . 21
1.11.4 Approaches to Transfer Learning . . . 21
1.11.5 Applications . . . 23
1.12 Related works . . . 23
1.12.1 Deep learning approaches . . . 23
1.12.2 Transformer-based Approaches . . . 24
1.13 conclusion . . . 25

2 Researach Methodology **27**
2.1 Research Framework . . . 27
2.1.1 Preparing Dataset . . . 27
2.1.2 NLP Processing . . . 28
2.1.3 Text Representation . . . 28
2.1.4 Model Building . . . 28
2.1.5 Grading Model . . . 28
2.1.6 Predictor Score . . . 28
2.1.7 Model Evaluation . . . 28
2.1.8 Effectiveness . . . 28
2.2 Design and steps . . . 28
2.2.1 Loading and Pre-processing Data . . . 30
2.2.2 Splitting the Dataset . . . 30
2.2.3 Sentence Embedding . . . 31
2.2.4 Grade predictor . . . 38
2.2.5 Evaluation . . . 39
2.3 Upload the model . . . 40

3 Implementation and Results **43**
3.1 Environment . . . 43
3.1.1 TensorFlow . . . 43
3.1.2 Pytorch python library . . . 43
3.1.3 Scikit-learn python library . . . 43
3.1.4 Hugging Face . . . 43
3.1.5 nlpaug . . . 44
3.2 Hardware used . . . 44
3.3 Dataset . . . 44
3.4 Splitting the data . . . 45
3.5 Augmentation . . . 45
3.6 BERT Results . . . 47
3.7 Siamese Bidirectional LSTM Results . . . 54
3.7.1 Comparison . . . 57
3.8 ELMO Results . . . 58
3.8.1 Comparison with other works . . . 59
3.9 Analysis and discussion . . . 60

List of Figures

1.1 Massive Open Online Courses Rising 5
1.2 Text similarity measurement 9
1.3 The variety of methods used in embeddings 12
1.4 Artificial intelligence 13
1.5 supervised-learning 14
1.6 unsupervised learning 15
1.7 deep learning 16
1.8 DEEP LEARNING MODELS 16
1.9 Transfer learning process 20

2.1 A pipeline for automated short answer scoring 27
2.2 Steps for automated short answer scoring 29
2.3 Steps required to build our model 29
2.4 Data prepossessing 30
2.5 Sentence list 30
2.6 BERT Architect pipeline 31
2.7 Concatenation 32
2.8 Tokenization 33
2.9 Token Conversion 33
2.10 LSTM Model Architecture 35
2.11 input preparation 36
2.12 Siamese LSTM Mode 36
2.13 ELMO Architecture 38
2.14 Embedding elmo 38
2.15 supervised-learning 39

3.1 Data's example 44
3.2 Distribution of answers by correctness 46
3.3 Distribution of Average Scores 46
3.4 Accuracy over epochs 47
3.5 Accuracy over epochs 48
3.6 Training validation loss 48
3.7 Training validation loss 49
3.8 Table of metric 50
3.9 Predected scores 50
3.10 Table of metric add quantize prediction 51
3.11 Human/Predicted Score 51
3.12 Human/Predicted Score 52
3.13 maximum length 54
3.14 maximum length 55
3.15 score vs predicted-score 55
3.16 score using quantize 56
3.17 score using quantize 56
3.18 predicted score 58
3.19 predicted score 58

List of Tables

3.1 Splitting the data 45
3.2 Splitting the data after augmentation 45
3.3 Summary of Research on Automatic Short Answer Grading 53
3.4 Performance Metrics 54
3.5 Updated Performance Metrics 55
3.6 Summary of Model Evaluations on the Mohler Dataset 57
3.7 Performance Metrics 58
3.8 Summary of Evaluation Metrics for Different Models 59
3.9 Summary of ASAG System Performance Using Deep Learning Models . . 59
3.10 Summary of Paragraph Embedding Methods and Performance Metrics . 60
3.11 Comparative Performance Metrics of NLP Models 60

General introduction

General introduction

With the spread of the Internet and technological evolution, teaching methods have also evolved. Traditionally, students had to come to the classroom to attend lectures and perform different exams. Now, teaching is also available through online platforms. MOOCs (Massive Open Online Courses) are a type of online platform that provides courses to a large number of students. The evaluation phase is an important task in the teaching process. However, it is difficult to be managed as in the traditional way. MOOCs allow for the evaluation of students through quizzes, which consist of various question types. The most commonly used question type is multiple-choice. Furthermore, in different situations, teachers need to evaluate students through short answers. This type of answer poses a challenge for the machine to understand and compare the student's answer with the target answer.

Deep learning techniques have proven their efficiency in different domains. These techniques are essential for understanding the nuances and context of student responses. In recent years, deep learning has emerged as a powerful tool in the field of text comprehension. Deep learning models, such as neural networks, have shown a remarkable ability to understand and process natural language, making them suitable for automating the grading of short answers. By leveraging deep learning, it is possible to develop systems that not only assess the correctness of a student's response but also understand the underlying concepts and ideas being conveyed. Building on this foundation of machine reading comprehension, automated systems have increasingly been employed in educational contexts. Automated systems make it easier to implement formative assessments, which are designed to monitor student learning and provide ongoing feedback that can be used by instructors to improve their teaching and by students to improve their learning. This advancement in technology holds significant potential to enhance the scalability and efficiency of educational assessments in MOOCs and other online learning platforms.

Problem statement

The main problem addressed in this study is how a machine can understand the text with a high level of efficiency. Additionally, this study explores how deep learning and transformer models can contribute to detect the similarity between the target answer and the answer of a student in online evaluation. Another key question is to what extent the choice of dataset and its preprocessing might enhance the effectiveness of training. To validate the system, its score predictions will be statistically compared with experts' actual ratings to assess its effectiveness in evaluating students' answers.

Research Objective

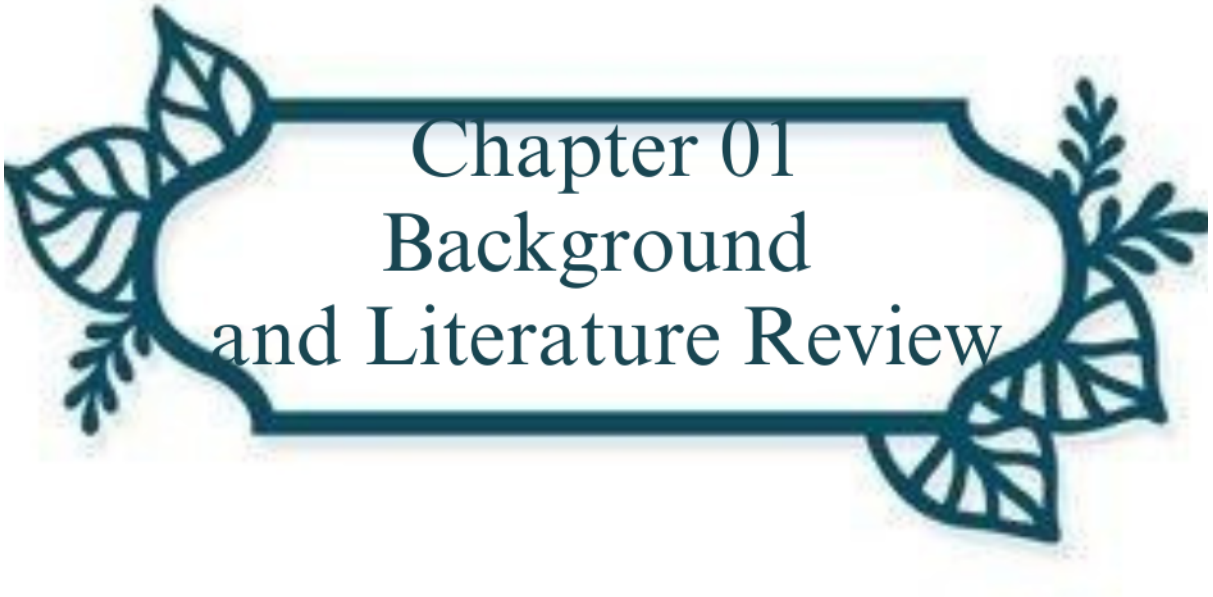
- **Objective:** Develop and validate an Automated Short Answer Grading (ASAG) system specifically tailored for the English language.
- **Target Audience:** Support university students in Algeria.
- **Methodology:** Utilize deep learning techniques to create the ASAG system.

- **Evaluation Strategy:** Statistically compare the ASAG system's score predictions with expert ratings to evaluate its effectiveness.
- **Research Aims:**
  - Assess the reliability of the ASAG system in evaluating students' reading comprehension skills.
- **Expected Outcomes:**
  - Improve the accuracy and efficiency of student performance assessments in distance education.

Thesis Organization

This thesis is organized into three main chapters:

1. **General Introduction:** Introduces the research topic, stating the problem and objectives of the study.
2. **Background and Literature Review:** This chapter reviews the relevant literature to provide a background and theoretical framework for the research.
3. **Methodology:** This chapter describes the research methodology, including data collection methods, tools used and Models architects,discussing the findings in detail
4. **Implementation and Results:** This chapter presents the environment work , Dataset, results of our models ,comparison with previous studies and discussing the results.
5. **General Conclusion:** provides a summary about this work including a description of the findings and perspectives for future research.



Chapter 01
Background and Literature Review

Chapter 1

Background and Literature Review

Introduction

Massive Open Online Courses (MOOCs) are online courses accessible worldwide, designed to provide flexible and affordable educational opportunities. The role of text comprehension in MOOCs is crucial as it enables personalized learning experiences, automates grading of assignments, and enhances content accessibility. By understanding and processing student inquiries and responses, these technologies can tailor educational content, provide immediate feedback, and convert course materials into more accessible formats, thereby enriching the learning environment for a diverse global audience. In This chapter presents a focused exploration into the specialized field of text comprehension using deep learning techniques. As we review the literature, we aim to understand how deep learning has revolutionized our ability to interpret, analyze, and generate human language. Text comprehension involves complex processes where machines are taught to understand and manipulate human language in a way that is both meaningful and contextually appropriate. Deep learning, with its advanced neural network architectures, has emerged as a powerful tool for handling the nuances and intricacies of language. These models excel in capturing latent features and dependencies within text, making them particularly effective for tasks ranging from simple classification to complex question answering and summarization.

1.1 Moocs definition

Massive Open Online Courses (MOOCs) are online learning programs accessible via the web, aiming to offer high-quality education to a wide global audience. These courses are usually free and open to all participants, embodying the fundamental principles suggested by their name MOOC.

As a whole, MOOCs must adhere to the fundamental criteria implicit in their name. **Massive**, signifies the broad reach these courses strive for, with some drawing in more than 150,000 learners and millions of users enrolled globally across multiple courses.The term "**Open**" indicates the four facets of openness—Reuse, Revise, Remix, and Redistribute—enabling individuals from any location, irrespective of age or income, and without prerequisites or course fees, to access educational materials. This openness also encompasses the incorporation of open educational resources (OER), such as PowerPoint presentations, video lectures, and assessments. **Online** underscores the ability to access MOOCs worldwide through the internet, facilitating both synchronous and asynchronous interactions among participants. Lastly, the component of **Courses** refers to the structured educational offerings within MOOCs, encompassing OER, clear learning objectives, networking tools, assessments, and learning analytics tools aimed at enhancing the educational experience [35].

The figure 1.1 [30] shows trends in Massive Open Online Courses (MOOCs), detailing supply and demand, student origins, and types of courses offered.

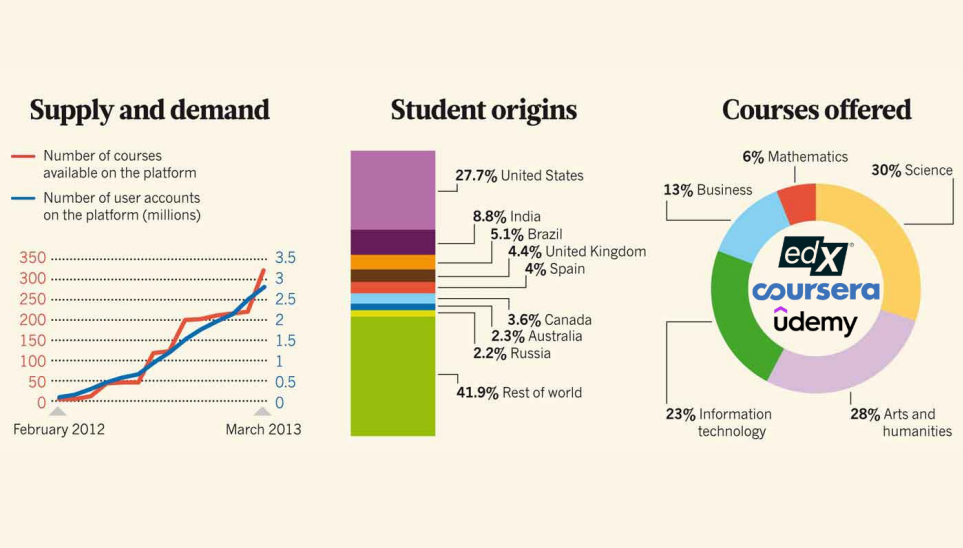


Figure 1.1: Massive Open Online Courses Rising

1.1.1 MOOCs forms

MOOCs are categorized into three primary forms [35],based on their structure and learning approach: Extended MOOCs, Connective MOOCs, and Blended MOOCs.

Extended MOOCs (xMOOCs):xMOOCs utilize a traditional knowledge transmission model, typically through video recordings of lectures or specially produced mini-lectures. These courses are structured linearly and focus on delivering knowledge while assessing learners through quizzes and exams. Pre-recorded video lectures from renowned educational institutions are a staple, supplemented by interactive elements like user forums and online chats. Prominent examples include Coursera's "Machine Learning" by Andrew Ng, edX's "Introduction to Computer Science" from Harvard, and Udacity's Nanodegree programs such as Data Science.

Connective MOOCs (cMOOCs): Based on the connectivist theory, cMOOCs emphasize networked learning and community-based knowledge sharing. These are learner-driven, encouraging exploration of personal interests and fostering connections among participants to create a collaborative learning environment. An example is "Change11," a forum for educators to discuss evolving educational practices.

BLENDED MOOCs (bMOOCs): Combining online and traditional face-to-face teaching methods, bMOOCs leverage the strengths of both formats. This hybrid approach enhances learning experiences by integrating MOOC technologies with conventional classroom interactions. It typically includes online lectures, readings, and assignments, alongside opportunities for peer interaction, group discussions, and hands-on projects. An illustrative case is the blended learning model adopted by the University of Yahya Fares, which merges online content with regular in-person sessions.

1.1.2 MOOCs Advantages

Massive Open Online Courses (MOOCs) provide unparalleled flexibility, enabling professors and students from diverse backgrounds to access educational resources and lectures at any time and from any location, without requiring prerequisites. These courses offer valuable learning opportunities to individuals who may not have access to traditional educational institutions. Many MOOCs are taught by faculty members from prestigious universities and colleges, expanding their reach and influence. While most courses are free of charge, there are paid options available for certificates or extra features, making high-quality education more accessible. MOOCs can

accommodate a large number of learners simultaneously and cover a wide range of subjects and disciplines. They also integrate interactive multimedia elements such as quizzes, assignments, and discussion forums, which greatly enhance the learning process.

1.2 MOOCs structure

Initially, MOOCs consisted mainly of pre-recorded lectures, study materials, and tasks that were exclusively available through university learning platforms. Yet, the quality of these recorded lectures has notably enhanced with time, now incorporating high-level animations and simulations that enhance engagement. The key elements of MOOCs generally encompass:

Video Lectures and Live Sessions: MOOCs utilize a range of presentation styles. The majority of the lectures are pre-recorded, enabling students to view them at their convenience. To enhance engagement and aid comprehension, these videos are often broken into short segments, usually lasting between 5 and 20 minutes. Besides the pre-recorded content, some MOOCs offer live sessions that promote real-time interaction. These sessions may include interactive polls, breakout rooms for group discussions, and direct Q and A opportunities with instructors.

Readings: Course materials may include textbooks, research papers, and articles from reputable sources, along with exclusive readings specifically prepared for enrolled students.

Discussion Forums: These forums are central to student and instructor communication. They provide a platform for students to ask questions and engage with their peers and the instructors. Moderation by volunteers or teaching assistants helps maintain productive and respectful interaction within these forums.

Assessments: Assessments play a vital role in the learning process within MOOCs, and their format varies extensively based on the course content. Options range from straightforward multiple-choice quizzes to more elaborate tasks such as programming assignments or written essays. To accommodate large numbers of participants, quizzes and objective tests are often auto-graded by automated systems, which provide quick feedback. Additionally, some courses incorporate peer reviews, allowing students to assess each other's work, fostering a collaborative learning environment [11].

1.3 Assessment

Assessment in education serves as a critical process for measuring and evaluating students' comprehension, skills, and knowledge acquired through learning activities. This evaluation is pivotal for educators as it enables them to gauge whether educational objectives are being achieved and empowers them to adapt teaching methodologies to better align with students' needs. Assessments encompass a variety of formats, including quizzes, assignments, and projects, each tailored to assess distinct aspects of learning. The insights derived from these assessments are indispensable for shaping future courses and implementing interventions aimed at improving student learning outcomes. Effective assessment practices play a pivotal role in fostering continual improvement and upholding educational standards [13] .

1.3.1 Phases of Assessment

Cuq outlines a four-stage process for assessment, each stage complementing the next: purpose, measurement, judgment, and decision.

Purpose: Here, the assessor identifies the goals, methods, and criteria for the evaluation.

Measurement: At this stage, the evaluator collects a comprehensive and relevant set of data to compare the learner's progress against expected outcomes.

Judgment: During this phase, the evaluator analyzes the collected data to assess the student's progress, ensuring alignment with the predetermined goals and objectives of the evaluation.

Decision: Based on the assessment, the evaluator determines the necessary actions to guide the educational or learning process in response to the learner's needs [13].

1.3.2 Types of Assessment in Education

Educational evaluations are broadly divided into two primary types: formative and summative assessments. Each type serves distinct purposes and functions within the learning process.

Formative Assessment Often referred to as "educative assessment," formative assessment is conducted throughout the learning process. It is not typically used for grading but is essential for providing feedback. Both teachers and students can engage in this feedback process to enhance learning and teaching practices respectively. Formative assessments are designed to monitor student progress and facilitate ongoing adjustments to both teaching methods and student learning strategies. Common examples include quizzes, draft reviews, and informal observations [7].

Summative Assessment This type of assessment is usually conducted at the conclusion of a learning period, such as at the end of a course, unit, or project, and is primarily used for grading purposes. Summative assessments aim to evaluate student comprehension and learning achievements through tests, final projects, or exams. They provide a means to gauge student learning outcomes at critical points, typically at the end of an instructional period [14].

Self-Assessment Self-assessment involves students evaluating their own learning. This method is integral to the competency-based educational approach, which places students at the core of the learning process. Through self-assessment, students engage actively in their learning journeys, reflecting on and evaluating their progress and skills [14].

1.3.3 Automated Grading Systems

Automated Grading Systems are advanced technological tools designed to assess and grade student assignments that require textual comprehension and analysis. These systems face the challenge of scoring student responses, which can vary widely in structure and complexity.

Educational assessments utilizing Automated Grading Systems typically handle two main types of questions: closed form and open answer questions.

Closed Form Questions: These questions have predetermined answers and require limited input from students, such as selecting the correct option. Examples include Multiple-Choice Questions (MCQs), True/False, Fill-in-the-Blanks, and Matching types. Due to their structured nature, automated systems can efficiently determine correct answers in these formats [34].

Open Answer Questions: These require more elaborate responses, allowing students to demonstrate their knowledge and reasoning freely. Formats include Short Answers, Essays, and Research Projects. Short answer questions, in particular, must meet several criteria:

They necessitate recalling external knowledge. Responses must be in natural language. The answer length ranges from a single word or phrase to one paragraph. The focus is on content rather than writing style [28].

Understanding these distinctions is vital for Automated Grading Systems to effectively evaluate such diverse responses.

1.3.4 Approaches to Automated Short Answer Grading (ASAG) Systems

This section outlines various methods utilized in ASAG systems to grade short answers effectively from [12]:

Concept Mapping: This technique dissects a student's answer into fundamental concepts for comparison with a model answer.

Information Extraction: Similar to extracting key features from audio samples in respiratory sound classification, this method extracts essential information from text responses. Techniques like Latent Semantic Analysis (LSA) assess the conceptual overlap between a student's response and the expected answer.

Corpus-based Methods: These methods use a large body of text to enhance understanding of context and semantics, akin to analyzing a dataset of audio recordings to detect respiratory conditions.

Machine Learning: Algorithms trained on datasets predict grading for new responses, similar to how machine learning models identify and classify respiratory disorders from training data.

Evaluation: In ASAG, competitions and shared datasets are crucial for benchmarking and refining systems, a practice that parallels the use of standardized datasets in your field to validate and advance diagnostic models.

Reference-based and Response-based Approaches: These techniques compare answers to a reference or analyze them independently to identify patterns, similar to methods used in medical audio analysis .

Hybrid Approaches: Combining reference-based and response-based methods enhances grading accuracy and reliability, mirroring how combining multiple diagnostic models can improve the prediction of respiratory disorders.

1.4 Text Comprehension

Text comprehension, the cognitive ability to derive meaning from written text, is a complex process that involves understanding the vocabulary employed within the text. It necessitates not only a grasp of the words but also the integration of background knowledge. This skill is extensively utilized across various contexts including educational settings, professional environments, and everyday life[24].

1.4.1 Text comprehension using deep learning

Text comprehension is a crucial task in various natural language processing (NLP) applications, aimed at understanding, analyzing, and interacting with language. The advent of deep learning technologies has significantly advanced the field of artificial intelligence, particularly in NLP. In the realm of comprehension, a key area within NLP, deep learning enables machines to understand and respond to queries based on unstructured text, thereby enhancing their ability to process and interpret language effectively[24].

1.5 Text Similarity Measurement

Text similarity measurement is a foundational concept in automatic short answer grading, playing a critical role in comparing target answers and student answers. The goal is to quantify how similar two pieces of text are, whether they are words, sentences, or entire documents.

Finding similarity between words is a fundamental part of text similarity which is then used as a primary stage for sentence, paragraph, and document similarities. Words can be similar in two ways: lexically and semantically, as the figure 1.2 shows :

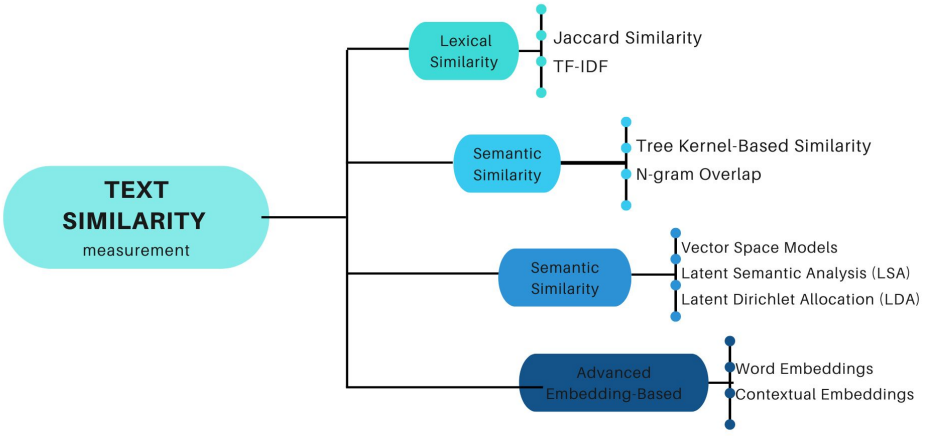


Figure 1.2: Text similarity measurement

1.5.1 Lexical Similarity

These methods focus primarily on the textual content without much emphasis on the meaning behind the words.

Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF enhances the Bag of Words (BoW) model by reducing the weight of more common words and increasing the weight of words that appear less frequently across documents. This helps differentiate texts based on unique terms. TF-IDF is a numerical statistic used to reflect how important a word is to a document within a collection or corpus. This statistic is used extensively in the fields of text mining and information retrieval.

A.Term Frequency (TF) The term frequency (TF) for a term t in a document d is defined as the number of times term t appears in document d, normalized by the total number of terms in document d:

$$\text{TF}(t, d) = \frac{\text{Number of times term } t \text{ appears in document } d}{\text{Total number of terms in document } d}$$

B.Inverse Document Frequency (IDF) The inverse document frequency (IDF) measures the importance of term t across a set of documents D. It is calculated by taking the logarithm of the ratio of the total number of documents to the number of documents containing the term t:

$$\text{IDF}(t, D) = \log\left(\frac{N}{|\{d \in D : t \in d\}|}\right)$$

where N is the total number of documents in the corpus, and $|\{d \in D : t \in d\}|$ is the number of documents where term t appears.

C.Combined TF-IDF Formula The TF-IDF score for a term t in a document d within a corpus D is calculated by multiplying the TF and IDF values:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D)$$

Jaccard Index

The Jaccard Index assesses the similarity between two sets based on their intersection and union. It is ideal for cases where the presence or absence of words is more important than their frequency or order. The formula is:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

1.5.2 Semantic Similarity Measures

Semantic methods evaluate the meaning behind the words, going beyond mere word presence or syntactic patterns.

Vector Space Models

Texts are represented as vectors in a high-dimensional space, typically using TF-IDF or word embeddings. Similarity is then calculated based on the distance or angle between these vectors.

A.Cosine Similarity Cosine similarity measures the cosine of the angle between two vectors, providing a measure of orientation independence:

$$\cos(\mathbf{v}_1, \mathbf{v}_2) = \frac{\mathbf{v}_1 \cdot \mathbf{v}_2}{\|\mathbf{v}_1\|\|\mathbf{v}_2\|}$$

B.Euclidean Distance The Euclidean distance is a measure of the true straight line distance between two points in Euclidean space:

$$d(\mathbf{v}_1, \mathbf{v}_2) = \sqrt{\sum_{i=1}^{n} (v_{1i} - v_{2i})^2}$$

Latent Semantic Analysis (LSA)

LSA uses singular value decomposition on a term-document matrix to reduce the dimensionality and capture hidden semantic structures.

Topic Models (Latent Dirichlet Allocation, LDA)

LDA analyzes texts based on their distribution over latent topics, which are inferred from the word distributions within the documents.

1.5.3 Advanced Embedding-Based Measures

These involve deep learning models that capture contextual nuances of text by considering the entire context in which words appear.

A.Word Embeddings (Word2Vec, GloVe) Generate dense vector representations of words based on their contextual usage in large corpora. Sentence or document similarity is often computed by aggregating these word vectors.

B.Contextual Embeddings (BERT, ELMo) These models provide word or sentence embeddings that dynamically adapt based on the text context, capturing semantic nuances and polysemy effectively.

1.5.4 Hybrid similarity

This method combines both syntactic and semantic approaches to capture a more comprehensive similarity measurement. For instance, hybrid models might use vector space models for semantic analysis and character-level comparisons for syntactic similarity. An example could be assessing the similarity between "fast car" and "quick vehicle." Semantically, both phrases describe a similar concept, and syntactically, "fast" and "quick," "car" and "vehicle" are synonyms but not identical strings."

1.6 Regression

Regression is a statistical method commonly utilized for two main objectives. First and foremost, it is extensively applied in forecasting and prediction, domains where it closely aligns with machine learning practices. Additionally, regression analysis can occasionally assist in uncovering causal connections between independent and dependent variables. Nevertheless, it is important to emphasize that regression on its own simply illustrates the associations between a dependent variable and multiple independent variables within a specific dataset[27].

1.6.1 Regression Models

Regression models are primarily used to predict the values of a dependent variable based on the values of one or more independent variables. In this discussion, we focus on linear regression and polynomial regression as the predictive models that best fit within the context[27].

A.Simple Linear Regression Simple Linear Regression involves a single independent variable to predict the dependent variable, following the equation $y = \beta_0 + \beta_1 x + \epsilon$. This model is useful for establishing a direct relationship between the two variables, emphasizing the impact of the independent variable without the interaction of other variables[27].

B.Multivariate Linear Regression (MLR) MLR predicts the outcome of a dependent variable using multiple explanatory variables. It aims to model the linear relationships between the independent variables x and the dependent variable y, represented by the equation $y = \beta_0 + \beta_1 x_1 + \cdots + \beta_m x_m + \epsilon$. This formulates a comprehensive framework that accounts for various factors simultaneously[27].

C.Polynomial Regression Polynomial Regression extends the linear model by considering the nth degree polynomial relationship between the independent and dependent variables. This approach allows for a more nuanced model that captures the curvilinear interactions between variables. The polynomial regression model can be expressed as $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_h x^h + \epsilon$, where h denotes the degree of the polynomial. This model is a special case of MLR, tailored to fit non-linear relationships more accurately[27].

1.7 Embedding

An embedding is a representation of words, phrases, or sentences as vectors of real numbers. These vectors are designed to capture semantic relationships between words based on their usage in context.

An embedding layer turns positive integers (indexes) into dense vectors of fixed size chosen from a pre-initialized matrix. For an integer-encoded text, the dense vector corresponding to those integers is selected.

have tried to compute embedding that capture the semantics of word sequences (phrases, sentences, and paragraphs), with methods ranging from simple additional composition of the word vectors to sophisticated architectures such as convolutions neural networks and recurrent

neural networks.Refer to Figure 1.3 for a detailed breakdown of these embedding types and their specific methods.

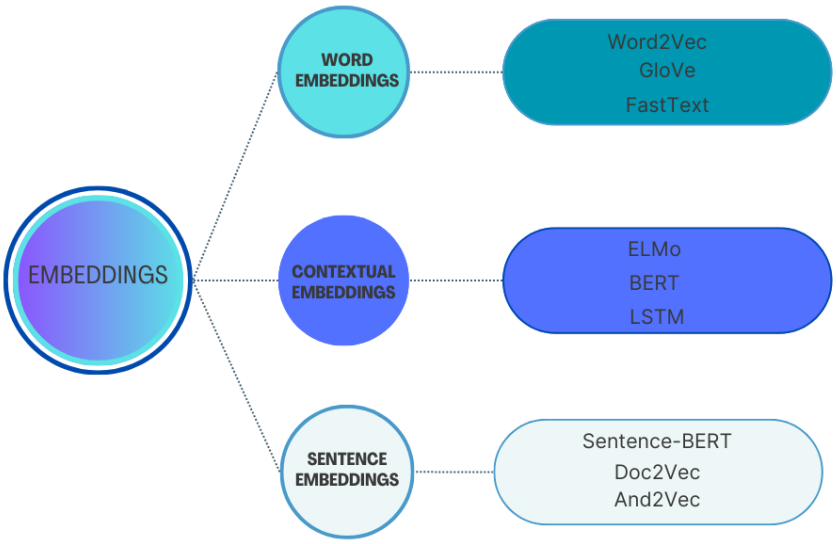


Figure 1.3: The variety of methods used in embeddings

1.8 Artificial intelligence

In popular culture, the term Artificial intelligence (AI) is often used as a catch-all word to mean any type of smart machine. However AI is overarching system , a large and complicated field that encompasses machine learning (ML) and deep learning. AI, is innovation dedicated to enabling machines to simulate human intellect and problem-solving capabilities , thereby handling complex tasks . AI stands for the development of the computer from being merely an instrument simulating human intelligence to its attaining intellectual and self-learning processes.As illustrated in Figure 1.4, AI forms the broadest category of this intellectual framework.

Figure 1.5 [3] visually encapsulates this process.

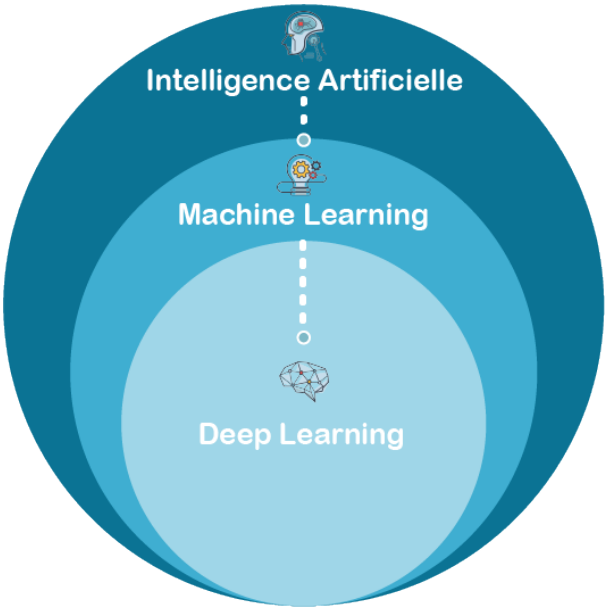


Figure 1.4: Artificial intelligence

1.9 Machine learning

machine learning has been a central concern in artificial intelligence since the early days when the idea of “self-organizing systems” was popular, the limitations inherent in the early neural network approaches led to a temporary decline in research volume. Machine learning, which is AI based on the notion that robots ought to be able to learn and adapt via experience, is AI that can automatically adapt with little assistance from humans.

A branch of artificial intelligence called ”machine learning” deals with teaching algorithms to use data to forecast or make judgments. With machine learning (ML), computer scientists can manually code each ”decision” to train a machine how to react to different sets of instructions rather than explicitly writing rules. This goes under the moniker ”train.” The machine analyzes the data and makes deductions based on a set of algorithms. (The computer can get more adept at carrying out a task or reaching a decision the more data it parses.)

1.9.1 Significance of Machine Learning

Machine Learning is one of the most popular sub-fields of Artificial Intelligence .In this era of abundant data it has lot of practical applications that help people to save money and time. From automating decision-making where machine analyze vast datasets and find insights in data without explicitly being programmed to lead increased efficiency and reduced human errors, this making it more efficient and adaptive, and allowing it handle complex and and dynamic environment and tasks. To increase the capabilities of humans in various fields whereby ML is permitting individuals to get things done more rapidly and effectively, support the improvement of modern item where numerous driving companies make ML a central portion of their operations and fostering sustainable solutions.

1.9.2 Type of Machine Learning

Supervised Machine Learning

Supervised learning is a machine learning technique that involves training a model on a labeled dataset, which contains input-output pairs. The goal is to learn a function that maps inputs to outputs. This learned function can perform two main tasks: regression, where it predicts a continuous value, or classification, where it predicts a class label for the input object. The essential task for a supervised learner is to generalize from the training data to unseen situations in a "reasonable" way, meaning the model should accurately predict the output for new, unseen inputs based on the patterns it has learned from the training data.

This technique requires the algorithm to be trained with examples that include both the inputs and the desired outputs, thereby enabling the model to infer the underlying function. After sufficient training, the learner should be capable of generalizing from this presented data to unseen examples [9]. The figure 1.5 [2] show how supervised-learning work :

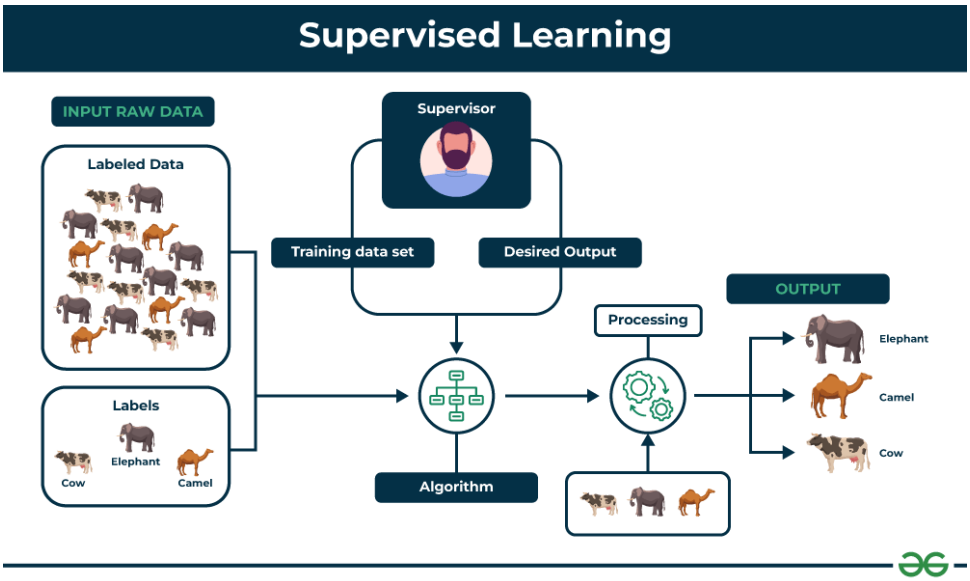


Figure 1.5: supervised-learning

Unsupervised learning

Unsupervised learning seems much harder: the goal is to have the computer learn how to do something that we don't tell it how to do! There are actually two approaches to unsupervised learning. The first approach is to teach the agent not by giving explicit categorizations, but by using some sort of reward system to indicate success. Note that this type of training will generally fit into the decision problem framework because the goal is not to produce a classification but to make decisions that maximize rewards. This approach nicely generalizes to the real world, where agents might be rewarded for doing certain actions and punished for doing others(classification and clustering)[10].

Figure 1.6 visually encapsulates this process [2].

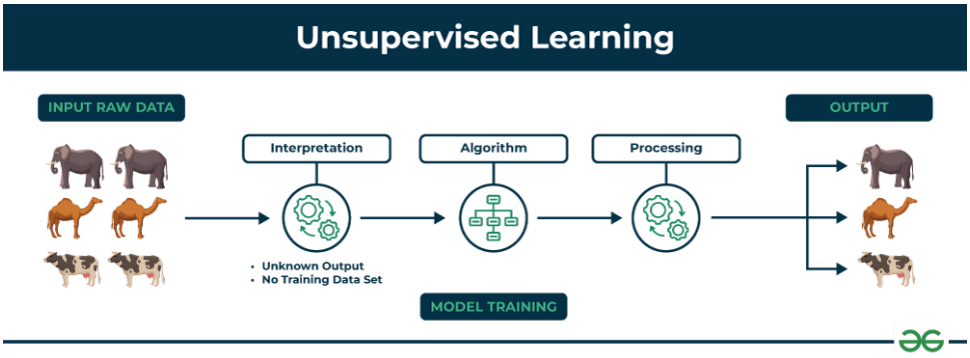


Figure 1.6: unsupervised learning

Semi-supervised learning

Semi-Supervised learning is a machine learning algorithm that works between the supervised and unsupervised learning so it uses both labelled and unlabelled data. It's particularly useful when obtaining labeled data is costly, time-consuming, or resource-intensive. This approach is useful when the dataset is expensive and time-consuming. Semi-supervised learning is chosen when labeled data requires skills and relevant resources in order to train or learn from it.

We use these techniques when we are dealing with data that is a little bit labeled and the rest large portion of it is unlabeled. We can use the unsupervised techniques to predict labels and then feed these labels to supervised techniques. This technique is mostly applicable in the case of image data sets where usually all images are not labeled [10] .

1.10 Deep learning

Deep Learning is a sub-field of machine learning that is inspired by the structure and function of the human brain,The concept underpinning this technology is very similar to how our brains function (biological neural networks). We compare new information with known information to arrive at a conclusion, called artificial neural networks,they can be used to solve a wide variety of problems, including image recognition, natural language processing, and speech recognition.it uses multi-layered involves a deeper level of automation in predictive

Today Deep learning AI has become one of the most popular and visible areas of machine learning, due to its success in a variety of applications, such as computer vision, natural language processing, and Reinforcement learning analytics[32].

1.10.1 Benefits of Deep learning

Deep learning models excel at automatically identifying the features necessary for classification or prediction tasks, eliminating the need for time-consuming manual feature engineering that requires domain expertise. This capability to extract new features from training data simplifies the modeling process, making it particularly valuable for complex tasks like image recognition. Furthermore, these models can handle large and intricate datasets with high dimensionality, which traditional machine learning algorithms may struggle with, enhancing their effectiveness in insight-driven tasks. Additionally, deep learning models offer versatility and adaptability through techniques such as transfer learning, where models can be fine-tuned for new tasks, making them suitable for dynamic environments. They also benefit from technological advancements that allow for faster training using GPUs (Graphics Processing Units) and TPUs (Tensor Processing Units), either locally or via distributed computing, accelerating the development and

deployment of deep learning solutions[32]. As depicted in the accompanying figure 1.7 [1], the performance of deep learning models significantly improves as the amount of data increases.

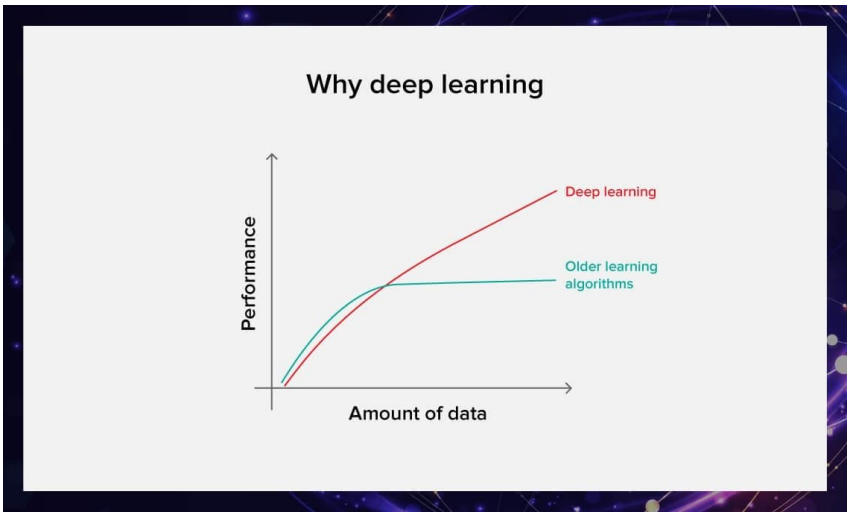


Figure 1.7: deep learning

1.10.2 DEEP MODELS in text comprehension description

In recent years, deep learning models have shown impressive achievements in voice recognition and computer vision. A large portion of the work in natural language processing with deep learning techniques has been using neural language models to learn word vector representations and then conducting composition over the learnt word vectors for categorization.

We can divide this models to categories as the figure 1.8 show :

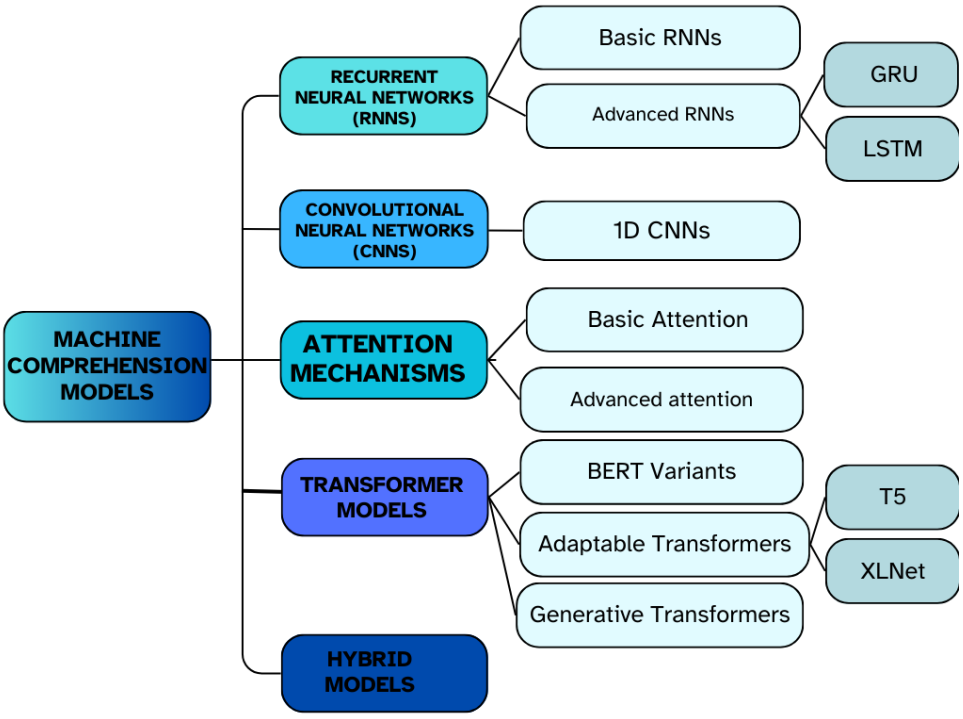


Figure 1.8: DEEP LEARNING MODELS

First category is Convolutional neural network (CNN)

Text-specific CNNs (1D CNNs) Effective for capturing local and positional patterns in text data, typically used for sentence classification tasks.

Convolutional Neural Networks is a prominent deep learning algorithm extensively utilized across various domains such as natural language processing (NLP), speech processing, and computer vision. Drawing inspiration from biological neural networks, the structure of CNNs mirrors the neural architecture found in the brains of animals and humans. More specifically, it emulates the visual cortex of a cat's brain, which consists of a sophisticated arrangement of cells.

CNNs for text analysis work by applying convolutional layers to text data, treating words or phrases similarly to how pixels in an image are treated. This approach allows the model to capture local dependencies and patterns in the text, such as n-grams (sequences of 'n' words).

For an n-words input text $w_1, \ldots, w_n$ we embed each symbol as a d-dimensional vector, resulting in word vectors $w_1, \ldots, w_n \in \mathbb{R}^d$. The resulting $d \times n$ matrix is then fed into a convolutional layer where we pass a sliding window over the text. For each l-words ngram:

$$u_i = [w_i, \ldots, w_{i+l-1}] \in \mathbb{R}^{d \times l}; \quad 0 \leq we \leq n - l$$

And for each filter $f_j \in \mathbb{R}^{d \times l}$ we calculate $\langle u_i, f_j \rangle$. The convolution results in matrix $F \in \mathbb{R}^{n \times m}$. Applying max-pooling across the ngram dimension results in $p \in \mathbb{R}^m$ which is fed into ReLU non-linearity. Finally, a linear fully- connected layer $W \in \mathbb{R}^{c \times m}$ produces the distribution over classification classes from which the strongest class is outputted. Formally:

$$u_i = [w_i; \ldots; w_{i+l-1}]$$

$$F_{ij} = \langle u_i, f_j \rangle$$

$$p_j = \text{ReLU}(\max_i F_{ij})$$

$$o = \text{softmax}(Wp)$$

In practice, we use multiple window sizes $l \in L$, $L \subseteq \mathbb{N}$ by using multiple convolution layers in parallel and concatenating the resulting p_l vectors. We note that the methods in this work are applicable for dilated convolutions as well[22].

Second category is Recurrent Neural Networks (Advanced RNNs)

are a type of deep learning model designed to handle sequential data. They are particularly well-suited for natural language processing tasks because of their ability to capture and model sequential dependencies in data.hey are powerful be- cause they have a high-dimensional hidden state with non- linear dynamics that enable them to remember and process past information. Furthermore, the gradients of the RNN are cheap to compute with back propagation through time. Unlike traditional neural networks, RNNs (Recurrent Neural Networks) leverage sequential information, a critical attribute in many scenarios where the structure embedded in the data sequence is informative. For instance, comprehending a word within a sentence often necessitates an understanding of its context, making RNNs analogous to short-term memory units comprising input layer x, hidden (state) layer s, and output layer y.

The standard RNN is formalized as follows: Given a sequence of input vectors $(x_1, \ldots, x_T)$, the RNN computes a sequence of hidden states $(h_1, \ldots, h_T)$ and a sequence of outputs $(o_1, \ldots, o_T)$ by iterating the following equations for $t = 1$ to T:

$$h_t = \tanh(W_{hx} x_t + W_{hh} h_{t-1} + b_h) \tag{1}$$

$$o_t = W_{oh} h_t + b_o \tag{2}$$

In these equations, W_{hx} is the input-to-hidden weight matrix, W_{hh} is the hidden-to-hidden (or recurrent) weight matrix, W_{oh} is the hidden-to-output weight matrix, and the vectors b_h and b_o are the biases. The undefined expression $W_{hh}h_{t-1}$ at time $t = 1$ is replaced with a special initial bias vector, h_{init}, and the tanh non linearity is applied coordinate-wise.

In the field, three deep RNN configurations have emerged:'Input-to-Hidden,'Hidden-to-Output,' and 'Hidden-to-Hidden.' These setups lead to a deeper RNN model that not only exploits the benefits of additional depth but also addresses the challenges of training deeper networks. A principal challenge with RNNs is their sensitivity to vanishing and exploding gradients—where gradients may diminish or expand exponentially due to the repeated multiplication of small or large derivatives throughout training. This gradient issue often results in the network's gradual forgetfulness of initial inputs as new data comes in.

To combat this, **Long Short-Term Memory (LSTM)** units have been developed. LSTMs enhance RNNs with memory blocks in their recurrent connections, each containing memory cells that preserve temporal states of the network. These blocks are equipped with gated units that meticulously regulate the flow of information, thus maintaining the network's ability to remember over longer periods without the decay typically seen in standard RNNs.

Gated Recurrent Units (GRUs) are another innovation designed to handle the challenges of traditional RNNs. GRUs streamline the structure seen in LSTMs by combining the forget and input gates into a single "update gate." This simplification allows GRUs to be more efficient computationally, while still retaining the ability to manage the flow of information without the vanishing gradient problem. GRUs provide a robustness comparable to LSTMs but are easier to modify and often faster to train, making them particularly useful in applications where computational resources are a constraint.

Additionally, the introduction of residual connections in very deep networks presents a significant advancement in alleviating the vanishing gradient issue. These connections enable direct pathways for gradients during back propagation, ensuring that even layers deep within the network can adjust their parameters effectively during training. This technique and its implications for deep learning architectures are explored further in Section [29]

Third category is Attention Mechanism

The attention mechanism allows to model the dependencies of words and their importance for the prediction task at a longer range in sentences. The modeling does not take into account the sequentially of words explicitly. An attention mechanism is an Encoder-Decoder kind of neural network architecture that allows the model to focus on specific sections of the input while executing a task. It dynamically assigns weights to different elements in the input, indicating their relative importance or relevance. By incorporating attention, the model can selectively attend to and process the most relevant information, capturing dependencies and relationships within the data. This mechanism is particularly valuable in tasks involving sequential or structured data, such as natural language processing or computer vision, as it enables the model to effectively handle long-range dependencies and improve performance by selectively attending to important features or contexts.

Recurrent models of visual attention use reinforcement learning to focus attention on key areas of the image. A recurrent neural network governs the peek network, which dynamically selects particular locations for exploration over time. In classification tasks, this method outperforms convolutional neural networks. Additionally, this framework goes beyond image identification and may be used for a variety of visual reinforcement learning applications, such as helping robots choose behaviours to accomplish particular goals. Although the most basic use of this strategy is supervised learning, the use of reinforcement learning permits more adaptable and flexible decision-making based on feedback from past glances and rewards earned throughout the learning process.[20]

Fourth category is Transformer Models

- **BERT Variants:** The most notable among transformer models, BERT (Bidirectional Encoder Representations from Transformers), developed by Google, revolutionized NLP pre-training. BERT utilizes a bidirectional approach to process text, enhancing understanding and prediction by considering the full context of a sentence. It is pre-trained on a substantial corpus, including the Books Corpus and Wikipedia, through two primary tasks: masked language model (MLM) and next sentence prediction (NSP). The architecture of BERT includes variants like BERT Base, with 12 transformer layers, and BERT Large, with 24 layers. BERT's comprehensive model integrates token, segment, and positional embeddings, enriching its input handling. The MLM task involves masking 15% of the words in each sequence to focus the training on context understanding rather than mere position-based memorization.

- **Skip-Thought Vectors:** The Skip-Thought vector model excels in embedding sentences into a numerical space that reflects sentence relationships. It employs operations such as element-wise product and absolute difference to capture interactive features and magnitudes of differences between sentence pairs. These operations help in semantic relatedness and paraphrase detection, showcasing the model's capability to encode both content and relational context.

- **Adaptable Transformers:** Adaptable Transformer models like T5 (Text-to-Text Transfer Transformer) and XLNet demonstrate flexibility across a range of tasks with minimal adjustments. These models are designed to facilitate effective knowledge transfer across different domains, exemplifying the adaptability essential for modern NLP challenges.

- **XLNet** is an advanced natural language processing model based on a bidirectional autoregressive approach, enhancing context understanding by pre-training on potential permutations of context words surrounding a target word. This model integrates key features from Transformer-XL, such as permutation language modeling and relative positional encoding, to handle longer texts and sequence dependencies efficiently. XLNet is available in two configurations: XLNet Base, with 12 layers, 12 attention heads, and 768 neurons, and XLNet Large, which features 24 layers, 16 attention heads, and 1024 neurons. Both models are primarily available in cased versions to accommodate the nuances of text case sensitivity, with the Base model often preferred for environments with limited computing resources like Google Colab.[18]

- **T5**, or the Text-to-Text Transfer Transformer, is a versatile deep learning model developed by Google researchers that redefines the approach to handling natural language processing (NLP) tasks. Distinctively, T5 converts all NLP problems into a text-to-text format, meaning that every task—be it translation, summarization, question answering, or even classification—is treated as a problem of generating some text from some other text. This approach simplifies the process of using the model across different tasks by maintaining a uniform system architecture and training process.

  The core idea behind T5 is to leverage the powerful Transformer architecture, specifically employing an encoder-decoder framework. This allows the model to process input text through its encoder, and then generate output text using the decoder. The extensive pre-training of T5 involves a large corpus drawn from diverse sources to cover a broad spectrum of language usage and styles. This pre-training is conducted using a "masked language model" objective, similar to BERT, but T5 extends this by training to predict one part of a sentence from another part, rather than just masked words.

  For fine-tuning, T5 adapts to specific tasks by applying task-specific prefixes (e.g., "translate English to German:") that signal to the model what kind of task it needs to perform,

using the same text-to-text format. This method enables the model to shift seamlessly between different types of tasks without significant modifications to its architecture, making it highly efficient for deployment in varied applications.[31]

1.11 Transfer learning

1.11.1 What Is Transfer Learning?

Definition 1. (Domain) A domain D is composed of two parts, i.e., a feature space X and a marginal distribution $P(X)$. In other words, $D = \{X, P(X)\}$. And the symbol X denotes an instance set, which is defined as $X = \{x|x_i \in X, i = 1, \ldots, n\}$.

Definition 2. (Task) A task T consists of a label space Y and a decision function f, i.e., $T = \{Y, f\}$. The decision function f is an implicit one, which is expected to be learned from the sample data.

Some machine learning models actually output the predicted conditional distributions of instances. In this case, $f(x_j) = \{P(y_k|x_j)|y_k \in Y, k = 1, \ldots, |Y|\}$.

In practice, a domain is often observed by a number of instances with/without the label information. For example, a source domain D_S corresponding to a source task T_S is usually observed via the instance-label pairs, i.e., $D_S = \{(x, y)|x_i \in X_S, y_i \in Y_S, i = 1, \ldots, n_S\}$; an observation of the target domain usually consists of a number of unlabeled instances and/or limited number of labeled instances [36].As illustrated in Figure 1.9 [15] , the principles of domains and tasks are crucial in the process of transfer learning, where pre-trained models from one domain (source domain) are adapted to new, but related tasks (target domain), significantly improving the learning efficiency and model performance by leveraging learned features.

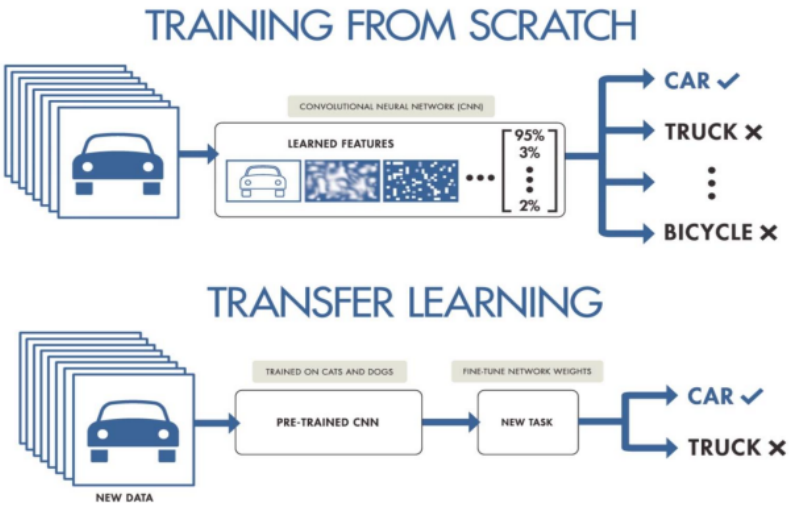


Figure 1.9: Transfer learning process

Transfer learning focuses on transferring the knowledge across domains, is a promising machine learning methodology for solving the problem. it is a powerful technique in machine learning that involves taking a model developed for one task and reusing it as the starting point for a model on a second task. According to the generalization theory of transfer, as proposed by psychologist C.H. Judd, learning to transfer is the result of the generalization of experience. It is possible to realize the transfer from one situation to another,as long as a person generalizes his experience. According to this theory, the prerequisite of transfer is that there needs to be a connection between two learning activities. In practice, a person who has learned the violin can

learn the piano faster than others, since both the violin and the piano are musical instruments and may share some common knowledge. The general idea is to use the knowledge a model has learned from a task with a lot of available labeled training data in a new task that doesn't have much data. Instead of starting the learning process from scratch, we start with patterns learned from solving a related task. Transfer learning This approach is particularly popular in the field of deep learning due to the large datasets and substantial computational resources typically required to train deep neural networks from scratch.

This approach leverages the knowledge learned by the pre-trained model on the original task and applies it to the new task, thereby reducing the amount of data and computational resources required to train the model for the new task.[37] Transfer learning is mostly used in computer vision and natural language processing tasks like sentiment analysis due to the huge amount of computational power required. Transfer learning isn't really a machine learning technique, but can be seen as a "design methodology" within the field.

1.11.2 Categories

Homogeneous transfer learning approaches are developed and proposed for handling the situations where the domains are of the same feature space. In homogeneous transfer learning, some studies assume that domains differ only in marginal distributions. Therefore, they adapt the domains by correcting the sample selection bias or co variate shift . However, this assumption does not hold in many cases. For example, in sentiment classification problem, a word may have different meaning tendencies in different domains. This phenomenon is also called context feature bias. To solve this problem, some studies further adapt the conditional distributions. Heterogeneous transfer learning refers to the knowledge transfer process in the situations where the domains have different feature spaces. In addition to distribution adaptation, heterogeneous transfer learning requires feature space adaptation, which makes it more complicated than homogeneous transfer learning [37].

1.11.3 Why Use Transfer Learning

According to DeepMind CEO Demis Hassabis, transfer learning is also one of the most promising techniques that could lead to artificial general intelligence (AGI) someday: "We think transfer learning is the key to general intelligence. And we think the key to doing transfer learning will be the acquisition of conceptual knowledge that is abstracted away from perceptual details of where you learned it from." By transferring knowledge from a well-trained model, we save time computational resources ,making it easier to fine-tune for the specific problem at hand and Improved Generalization as they have learned more generic features.

transfer learning facilitates multi-task learning, where a single model is trained to perform multiple related tasks simultaneously, improving performance and efficiency.Using pre-trained models can speed up the development process, allowing for quicker deployment of machine learning solutions.also Models can be continuously updated and fine-tuned with new data, improving their performance over time. [16]

1.11.4 Approaches to Transfer Learning

We will explore several strategies to leverage pre-existing deep learning models and techniques for effective model reuse and transfer learning [37].

- **Training a Model to Reuse it**

  Imagine you want to solve task A but don't have enough data to train a deep neural network. One way around this is to find a related task B with an abundance of data. Train the deep neural network on task B and use the model as a starting point for solving task A. Whether you'll need to use the whole model or only a few layers depends heavily on the problem you're trying to solve.

  If you have the same input in both tasks, possibly reusing the model and making predictions for your new input is an option. Alternatively, changing and retraining different task-specific layers and the output layer is a method to explore.

- **Using a Pre-Trained Model**

  The second approach is to use an already pre-trained model. There are a lot of these models out there, so make sure to do a little research. How many layers to reuse and how many to retrain depends on the problem.

  Keras, for example, provides numerous pre-trained models that can be used for transfer learning, prediction, feature extraction and fine-tuning. You can find these models, and also some brief tutorials on how to use them, here. There are also many research institutions that release trained models.

  This type of transfer learning is most commonly used throughout deep learning.

- **Feature Extraction**

  Another approach is to use deep learning to discover the best representation of your problem, which means finding the most important features. This approach is also known as representation learning, and can often result in a much better performance than can be obtained with hand-designed representation.

  In machine learning, features are usually manually hand-crafted by researchers and domain experts. Fortunately, deep learning can extract features automatically. Of course, this doesn't mean feature engineering and domain knowledge isn't important anymore — you still have to decide which features you put into your network. That said, neural networks have the ability to learn which features are really important and which ones aren't. A representation learning algorithm can discover a good combination of features within a very short time frame, even for complex tasks which would otherwise require a lot of human effort.

  The learned representation can then be used for other problems as well. Simply use the first layers to spot the right representation of features, but don't use the output of the network because it is too task-specific. Instead, feed data into your network and use one of the intermediate layers as the output layer. This layer can then be interpreted as a representation of the raw data.

  This approach is mostly used in computer vision because it can reduce the size of your dataset, which decreases computation time and makes it more suitable for traditional algorithms, as well.

- **Popular Pre-Trained Models**

  There are some pre-trained machine learning models out there that are quite popular. One of them is the Inception-v3 model, which was trained for the ImageNet "Large Visual

Recognition Challenge." In this challenge, participants had to classify images into 1,000 classes like "zebra," "Dalmatian" and "dishwasher."

Here's a very good tutorial from TensorFlow on how to retrain image classifiers.

Microsoft also offers some pre-trained models, available for both R and Python development, through the MicrosoftML R package and the Microsoftml Python package.

Other quite popular models are ResNet and AlexNet. we also encourage a visit to pre-trained.ml, a sortable and searchable compilation of pre-trained deep learning models complete with demos and code.

1.11.5 Applications

Transfer learning is extensively applied across various domains to enhance the efficiency and effectiveness of model development. It is employed in image recognition for tasks like facial and object recognition, in natural language processing for sentiment analysis and translation, and in speech recognition to accommodate diverse accents and dialects. Autonomous vehicles utilize it to adapt to changing driving conditions, while in healthcare, it helps fine-tune diagnostic models for specific medical conditions. In robotics, it reduces the need for extensive retraining by adapting models to new tasks. The technique is also critical in anomaly detection, financial services for fraud detection and trading, in climate modeling for adapting to different environmental conditions, and in education to create personalized learning experiences. These applications highlight transfer learning's pivotal role in leveraging existing knowledge to address new challenges efficiently [19].

1.12 Related works

Several approaches have been proposed in order for the aim of text comprehension and automatic short answer grading.There are multiple Auto scoring systems available for different languages. Fall into two broad categories. (1) Deep learning approaches (2)Transformer-based Approaches which minimized manual feature engineering and focus on the semantic information.

1.12.1 Deep learning approaches

- In This study [8], examines the use of deep learning to enhance the accuracy and efficiency of automatic short answer grading (ASAG) in blended learning environments. By delving into dataset curation and refining evaluation metrics, the research aims to scale grading processes and reduce human biases, offering practical guidelines for educators to integrate ASAG technologies effectively.

- The authors in [31] introduce a novel model using Siamese Long Short-Term Memory Networks (LSTMs) with Earth Mover's Distance (EMD) pooling to grade short answers. This approach aims to surpass traditional attention mechanisms by capturing the semantic nuances in student responses, significantly reducing grade estimation errors and improving the linguistic accuracy of ASAG systems. The system achieved a Pearson correlation coefficient of 0.7085 and a Mean Absolute Error (MAE) of 0.7009 when compared to the average scores of manual grading by instructors, indicating a strong linear relationship and reasonable error levels respectively.

- In the paper "On Deep Learning Approaches to Automated Assessment Strategies," [6] Abbirah Ahmed and colleagues explore deep learning applications in short answer grading, highlighting its role in processing natural language within educational technologies.

Their comprehensive review compares various models, discusses dataset and metric developments, and provides guidelines to educators for effectively using these technologies in automated assessments.

- In [21]This study develops an ASAG system using semantic networks and Support Vector Machines (SVM) to assess the semantic similarity in text-based answers. By extracting features from semantic networks to train the SVM, the approach achieves improved correlation and error metrics over previous methods, showcasing the potential of combining semantic analysis with machine learning for educational assessments.

- In the study "A deep-learning-based grading system (ASAG) for reading comprehension assessment using aphorisms as open-answer-questions," [26]Researchers use BERT and Skip-Thought models to grade short answers in a reading comprehension assessment. Adjusting various model hyperparameters, the study finds that BERT outperforms in understanding and processing natural language, enhancing the objectivity and efficiency of grading and highlighting the potential of deep learning in educational assessments.

- In the comprehensive survey [20]Haller et al. review significant advancements in ASAG, emphasizing the transition from hand-engineered features to representation learning via large language models like BERT. The survey discusses the integration of traditional and advanced techniques, marking a shift towards more reliable and efficient educational assessments facilitated by AI and machine learning technologies.

1.12.2 Transformer-based Approaches

- In [17],This study explores enhancing ASAG systems using paragraph vectors and transformer-based models to improve semantic similarity measures between student and model answers. By comparing pre-trained paragraph vectors and domain-specific fine-tuned models like Roberta-large and Scibert, the findings suggest that while paragraph vectors provide a strong baseline, domain-specific fine-tuning better addresses the unique language nuances in academic assessments, leading to improved grading accuracy.

- In [25], Luo and team investigate the effectiveness of BERT and Sentence BERT (SBERT) on the Short Answer Scoring V2.0 dataset for grading short, free-text responses from students. Their study finds that SBERT, particularly when fine-tuned for semantic similarity tasks, excels in grading shorter answers due to its superior handling of sentence embeddings, achieving higher accuracy and F1 scores compared to BERT, particularly in tasks involving shorter responses.

- In the study [23] ,the authors reviews advancements in ASAG through AI and NLP, highlighting a shift from traditional feature-engineering to using large language models like BERT for multilingual datasets. The paper discusses how deep learning models streamline grading by reducing preprocessing needs and improving accuracy close to human levels, demonstrating their effectiveness in large-scale educational settings.

- in the work [18], Ghavidel and colleagues use transformer-based models BERT and XLNET to automate grading of short answers, focusing on the SciEntBank dataset. They fine-tune these models to assess semantic relationships between student and model answers, achieving superior performance metrics across various tasks. Their results underscore the potential of using advanced models to bypass traditional manual feature engineering, enhancing grading accuracy and reliability in educational settings.

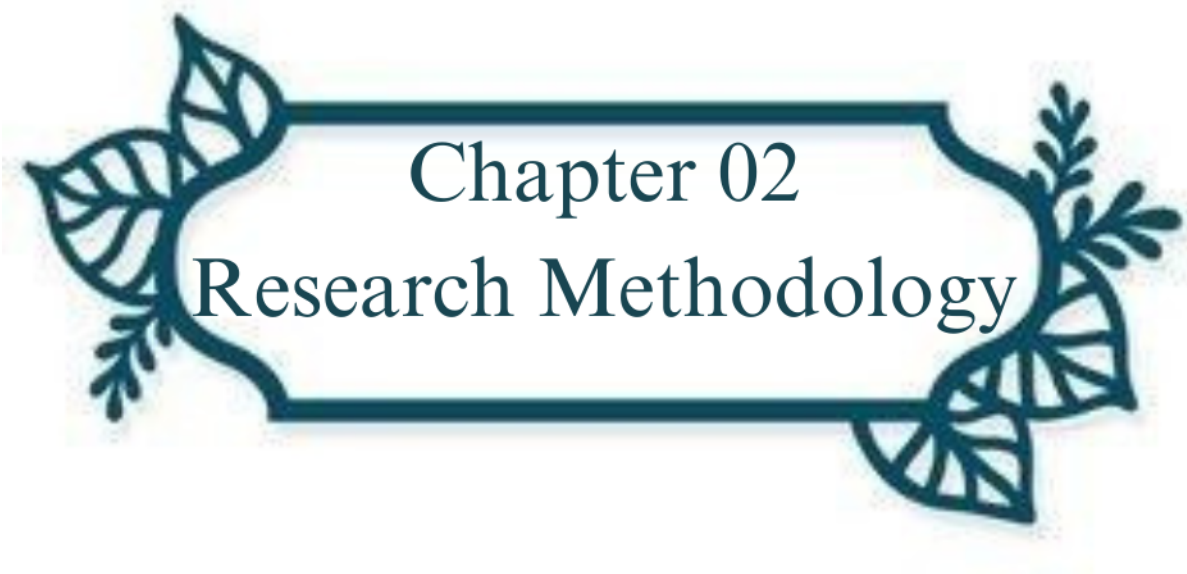
- In the study [34],Wilianto and team employ sentence-transformers to automate the grading of short answers in a high school e-learning environment. Their findings indicate that

the all-MiniLM-L6-v2 model outperforms other tested models in accuracy and processing speed, closely aligning with teacher-assigned grades. This study showcases the feasibility of using sentence-transformers to provide efficient, real-time grading in educational assessments, potentially easing the grading load on educators.

1.13 conclusion

In concluding our examination of text comprehension through deep learning within the context of Automated Short Answer Grading (ASAG), it is clear that deep learning has significantly transformed the educational technology landscape. We have witnessed remarkable advancements in ASAG facilitated by the deployment of sophisticated neural networks, such as Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and more recently, groundbreaking models like BERT and Elmo . These technologies have revolutionized educational assessment methodologies, enabling a depth of text comprehension that transcends mere keyword detection to provide detailed insights into student responses.

In the next chapter we describe our research methodology that aims to enhance the reliability and fairness of automated short answer grading systems.



Chapter 02
Research Methodology

Chapter 2

Researach Methodology

Introduction

In this section we will explore the methodology implemented to accomplish this project. The study will adhere to the depicted workflow process for the automatic evaluation of short answers. We will outline our comprehensive design and the techniques employed within it and assess the impact of machine learning on enhancing the outcomes previously achieved with the ASAG methodology.

2.1 Research Framework

As illustrated in the workflow diagram (Figure 2.1), our methodology for automated short answer scoring is systematically divided into several key phases, each critical to the development and evaluation of the scoring model

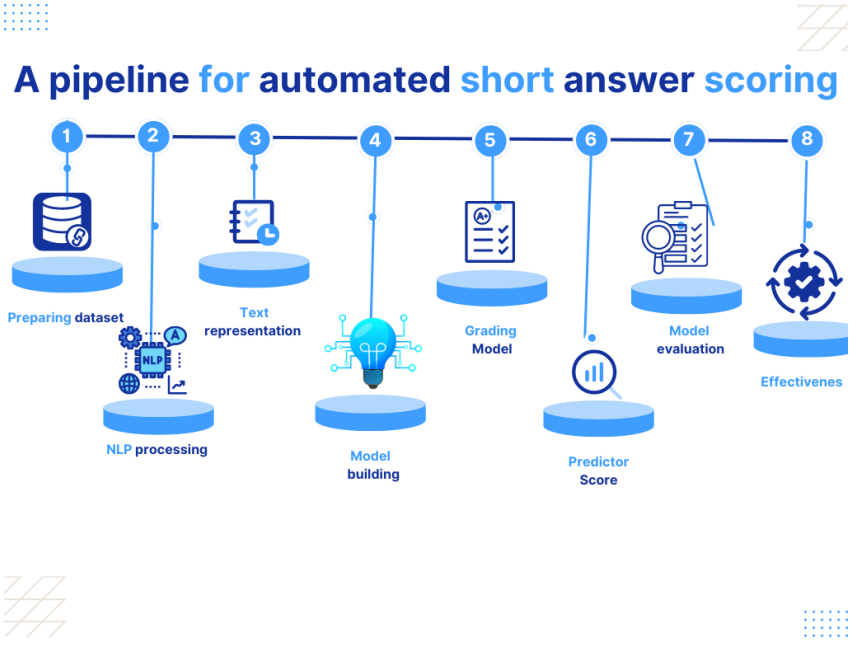


Figure 2.1: A pipeline for automated short answer scoring

2.1.1 Preparing Dataset

This initial phase involves the collection and preparation of the dataset that will be used for training the automated scoring system. It includes gathering short answers from various sources and possibly tagging them with expert scores. The data will be cleansed, normalized, and formatted to ensure consistency and quality for subsequent processing stages.

2.1.2 NLP Processing

After preparing the dataset, the next stage involves applying Natural Language Processing (NLP) techniques to refine and structure the text data for analysis. This crucial phase includes tokenization, where text is segmented into meaningful elements; lemmatization and stemming to reduce words to their root forms. These processes enhance the dataset's readiness for the subsequent machine learning stages by ensuring the text is clean, structured, and semantically nuanced.

2.1.3 Text Representation

At this stage, the prepared text data is transformed into a format suitable for machine learning models. This involves using Natural Language Processing (NLP) techniques to convert text into numerical or tokenized formats.

2.1.4 Model Building

This stage focuses on the development of the grading model itself. It involves selecting appropriate deep learning algorithms and configuring them to handle the specifics of short answer scoring.

2.1.5 Grading Model

The grading model applies the trained algorithms to new short answer submissions to predict scores. It must accurately reflect the standards set by human scorers and be robust enough to handle a variety of answer styles and complexities.

2.1.6 Predictor Score

Once the model is applied, it generates scores for each answer. This step converts the model outputs, into a final score that aligns with the scoring system used in educational settings.

2.1.7 Model Evaluation

In this critical phase, the performance of the grading model is assessed using statistical metrics such as Pearson's correlation coefficient, RMSE (Root Mean Squared Error), and Accuracy. These metrics evaluate how closely the predicted scores match the actual scores provided by human experts and are essential for validating the effectiveness of the model.

2.1.8 Effectiveness

The final phase involves analyzing the overall effectiveness of the automated scoring system. This includes evaluating its reliability, fairness, and adaptability to different types of questions and answers.

2.2 Design and steps

In this section, we will outline the steps taken to develop our model, from data pre-processing to model training. The accompanying chart, Figure 2.2 and Figure 2.3, provides an overview of the key methodology employed in our approach.

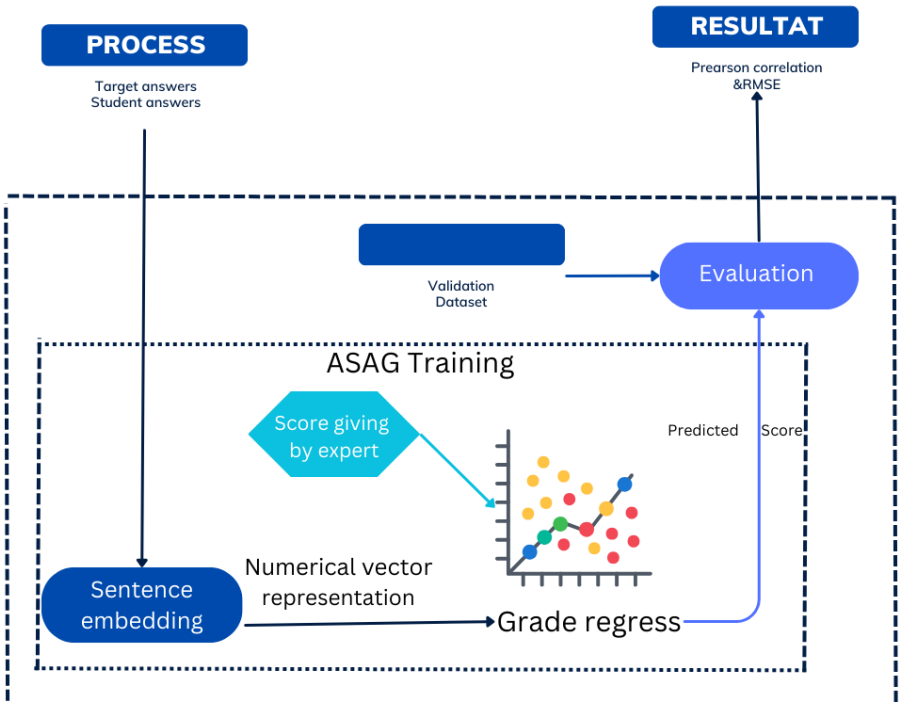


Figure 2.2: Steps for automated short answer scoring

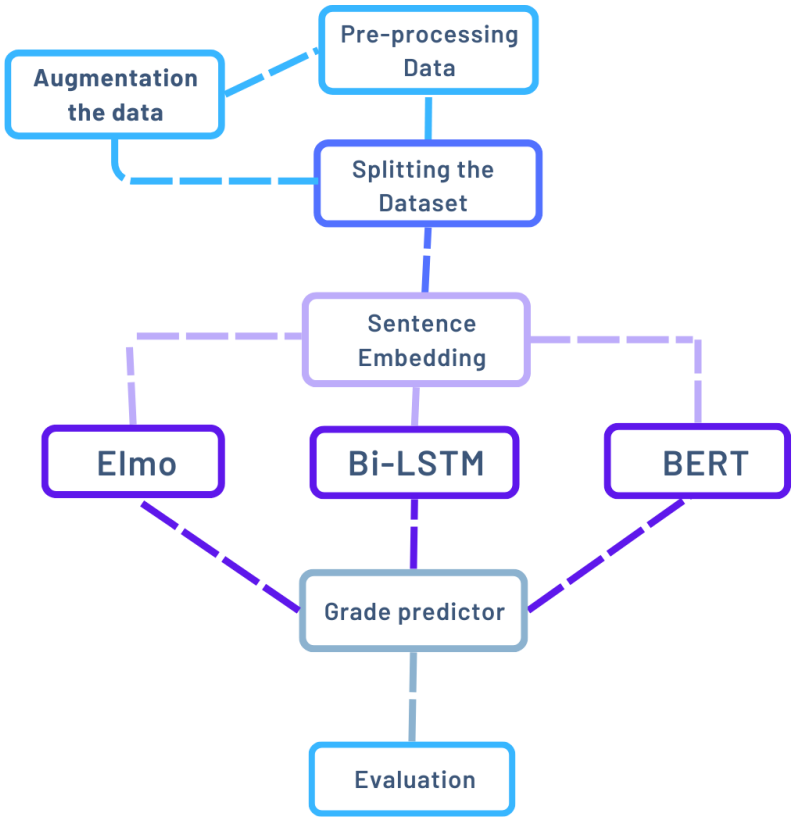


Figure 2.3: Steps required to build our model

2.2.1 Loading and Pre-processing Data

loaded the dataset from a Google Drive location into a pandas DataFrame and cleaned it by removing an extraneous column that appeared to be an accidental index addition. we then displayed the initial few rows for a quick verification of the data structure. Next, we separated the data into specific categories: questions, target answers, and student answers, which we cleaned by removing punctuation, and converting all text to lowercase for uniformity. These cleaned text strings, along with their corresponding labels (scores) stored in separate lists. This structured and cleaned dataset is crucial for the accuracy and efficiency of our subsequent machine learning tasks, ensuring that the input data is standardized and optimal for developing reliable models. As figure 2.4 show

Target Answers Sample: ['to simulate the behaviour of portions of the desired software product', 'to simulate the behaviour of portions of the desired software product', 'to simulate the behaviour of portions of the desired software product', 'to simulate the behaviour of portions of the desired software product', 'to simulate the behaviour of portions of the desired software product'']
Student Answers Sample: ['high risk problems are address in the prototype program to make sure that the program is feasible a prototype may also be used to show a company that the software can be possibly programmed', 'to simulate portions of the desired final product with a quick and easy program that does a small specific job it is a way to help see what the problem is and how you may solve it in the final project', 'a prototype program simulates the behaviors of portions of the desired software product to allow for error checking',]
Labels Sample: [3.5, 5.0, 4.0, 5.0, 3.0, 2.0, 2.5]

Figure 2.4: Data prepossessing

Additionally, we create list named conca-sentences by concatenate each answer with its corresponding target answer, separated by a space, to ensure the combination of relevant pairs. This step is to confirm the correctness of the pre-processing operations. like i do in the figure 2.5:

['high risk problems are address in the prototype program to make sure that the program is feasible a prototype may also be used to show a company that the software can be possibly programmed to simulate the behaviour of portions of the desired software product',
 'to simulate portions of the desired final product with a quick and easy program that does a small specific job it is a way to help see what the problem is and how you may solve it in the final project to simulate the behaviour of portions of the desired software product',
 'a prototype program simulates the behaviors of portions of the desired software product to allow for error checking to simulate the behaviour of portions of the desired software product',
 'defined in the specification phase a prototype stimulates the behavior of portions of the desired software product meaning the role of a prototype is a temporary solution until the program itself is refined to be used extensively in problem solving to simulate the behaviour of portions of the desired software product']

Figure 2.5: Sentence list

2.2.2 Splitting the Dataset

To ensure the model is evaluated impartially, we split the dataset into training and validation sets. This split helps in tuning the hyperparameters without overfitting to the training data. we allocat 20 to validation. The exact number of samples for each set is calculated based on the total size of the dataset. The splitting is performed randomly to prevent any biases that might occur if the data has a particular order.

Due to the initial small size of the dataset and a high concentration of similar scores,we implemented data augmentation techniques and adjusted the distribution to balance the dataset.

This step was crucial in enriching the dataset's diversity, allowing the model to perform better across a broader range of responses and improving the robustness of the results. Augmentation involved synthesizing new data points by slightly modifying existing answers and generating synthetic samples through techniques such as back-translation and paraphrasing. Balancing was achieved by oversampling under-represented classes to ensure a more uniform distribution of scores, which is essential for preventing model bias towards more frequent labels.

2.2.3 Sentence Embedding

To implement the sentence embedding, deep learning approaches are considered: BERT ,ELMO and Bi-LSTM were utilized. BERT is a model based on Transformers, essentially a mix of fully connected neural networks and mechanisms known as Self-Attentions. These allow it to capture significant text fragments related to a specific word, regardless of their order in the sequence. ELMO (Embeddings from Language Models) generates deep contextualized word representations by analyzing the entire sentence using bi-directional LSTM networks, capturing syntax and semantics effectively. LSTMs (Long Short-Term Memory networks) are a type of recurrent neural network designed to learn long-term dependencies in sequential data, making them suitable for tasks like language modeling and sequence prediction. Together, these models provide rich, context-aware embeddings that significantly enhance the performance of NLP tasks, the approach will be described in detail below:

A.BERT Architecture The BERT methodology is implemented to facilitates the retention of the text's original context and nuances.

we explored various text input strategies to enhance model performance by supplying a single concatenated sentence that includes both the Student Answer (ST) and the Target Answer (TA). This approach provides the flexibility to tailor the input format according to the specific context of the task. the figure 2.6 show the BERT Architecture

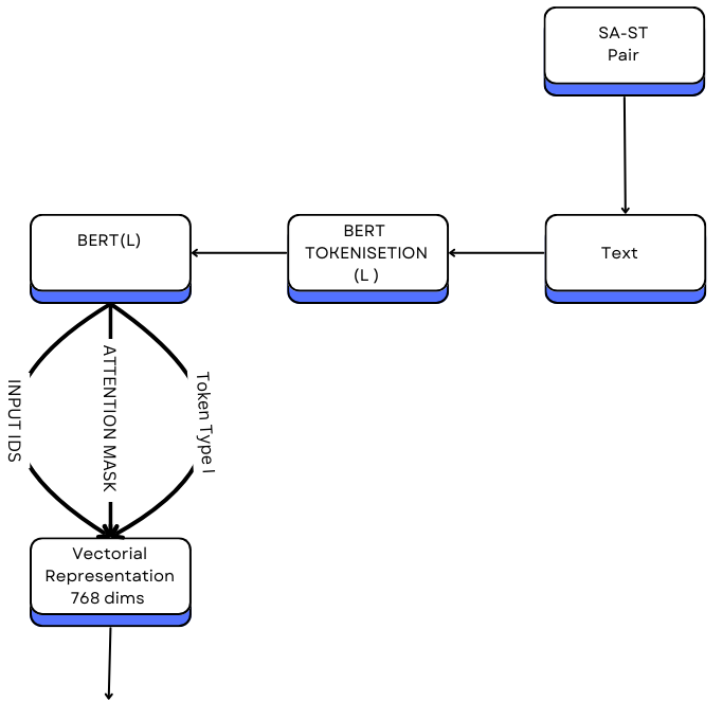


Figure 2.6: BERT Architect pipeline

- **Preprocessing Function for BERT**

  This function is designed to preprocess text data for compatibility with the BERT model. we start by importing and initializing the BertTokenizer with the pretrained "bert-base-uncased" model. This tokenizer is crucial for converting text into tokens that the BERT model can interpret.

  Like this example :

  Original Answer (Student's Answer): "To simulate portions of the desired final product with a quick and easy program that does a small specific job it is a way to help see what the problem is and how you may solve it in the final project."

  Model Answer (Target Answer):"To simulate the behaviour of portions of the desired software product."

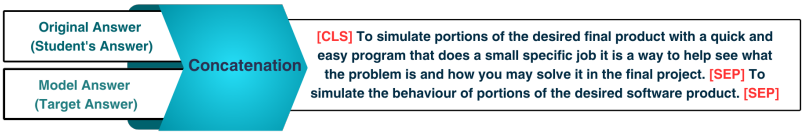


Figure 2.7: Concatenation

We then iterate over a list of pairs, each consisting of an answer and its corresponding target answer. For each pair,we tokenize the text, incorporating special tokens ([CLS] for the start and [SEP] for separation) to prepare the input for BERT processing.

Next,we determine the maximum length from these tokenized inputs to ensure uniformity across all inputs, which is vital for effective batching and model input handling.

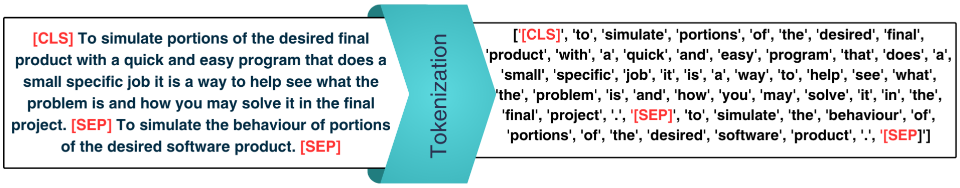


Figure 2.8: Tokenization

We convert each text pair into input IDs (token identifiers), token type IDs (to distinguish between the two sentences in a pair), and attention masks. These attention masks help the model to focus on the significant parts of the input and ignore the padding during both training and inference phases.

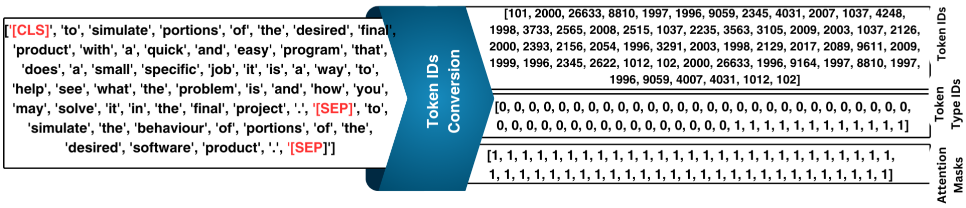


Figure 2.9: Token Conversion

After processing the text,we convert the outputs into PyTorch tensors to integrate seamlessly with PyTorch-based deep learning models.

We also handle additional tensor conversions, such as turning data indices into tensors, which might be useful for batch sample tracking or data management within a learning framework.

- **Setting up and Implementation Data Loaders**

After preprocessing the text data, all key components—input IDs, token type IDs, and attention masks—are converted into PyTorch tensors, making them suitable for neural

network processing. This step is essential for preparing the dataset for efficient utilization in machine learning workflows.

Subsequently, we combine these components into a single Tensor Dataset object. This aggregation includes the input IDs, attention masks, token type IDs, labels, and a tensor for indexing. This consolidated dataset structure facilitates streamlined batch processing and data handling during the training and inference stages of the neural network model.

- **BERT-parameters**

  In the implementation of our BERT-based sequence classification model,webegin by configuring data loaders with a batch size of 32. The training data loader employs a RandomSampler to select batches randomly, which ensures that the model encounters a diverse array of data during training. Conversely, the validation data loader utilizes a SequentialSampler for orderly data processing, which aids in consistent evaluation.

  We initialize the BERT model using the 'bert-base-uncased' pretrained configuration. This model is specifically set up for regression tasks, with num-labels set to 1.wealso configure the model to neither output attentions nor hidden states (output-attentions=False, output-hidden-states=False).

  For optimization,we use the AdamW optimizer, selected for its effectiveness in handling sparse gradients and its incorporation of weight decay. The learning rate is set at 2e-5 a default learning rate such as 5e-5 is typically suggested based on the model's original training and general efficacy across various tasks. However, it's common to adjust this rate to 2e-5 or similar values based on specific needs, like we try to use other rate but give a low results that's why we change it to 2e-5 to to get a good results . This adjustment can help address task-specific nuances, improve empirical outcomes, and ensure stability in training dynamics.These adaptations are crucial for optimizing model performance and stability in diverse application contexts, with an epsilon value of 1e-8 to improve numerical stability. Additionally, a learning rate scheduler featuring a linear warmup is implemented to gradually adjust the learning rate across 15 epochs. This adjustment is based on the total number of training steps, which are calculated from the number of batches and epochs.

  Throughout the training process,we set a seed value to ensure reproducibility. Each epoch involves monitoring the model's performance by tracking both training loss and validation accuracy. The validation phase involves evaluating the model against the validation set, where both accuracy and RMSE (Root Mean Square Error) are computed to ensure a robust assessment of performance.

B.Siamese LSTM Model Architecture The Siamese LSTM model is designed to assess the similarity between two sequences, which are typically text data in natural language processing tasks. This model architecture utilizes several key components to process and compare the sequences effectively. The figure 2.10 [33] show LSTM Model Architecture:

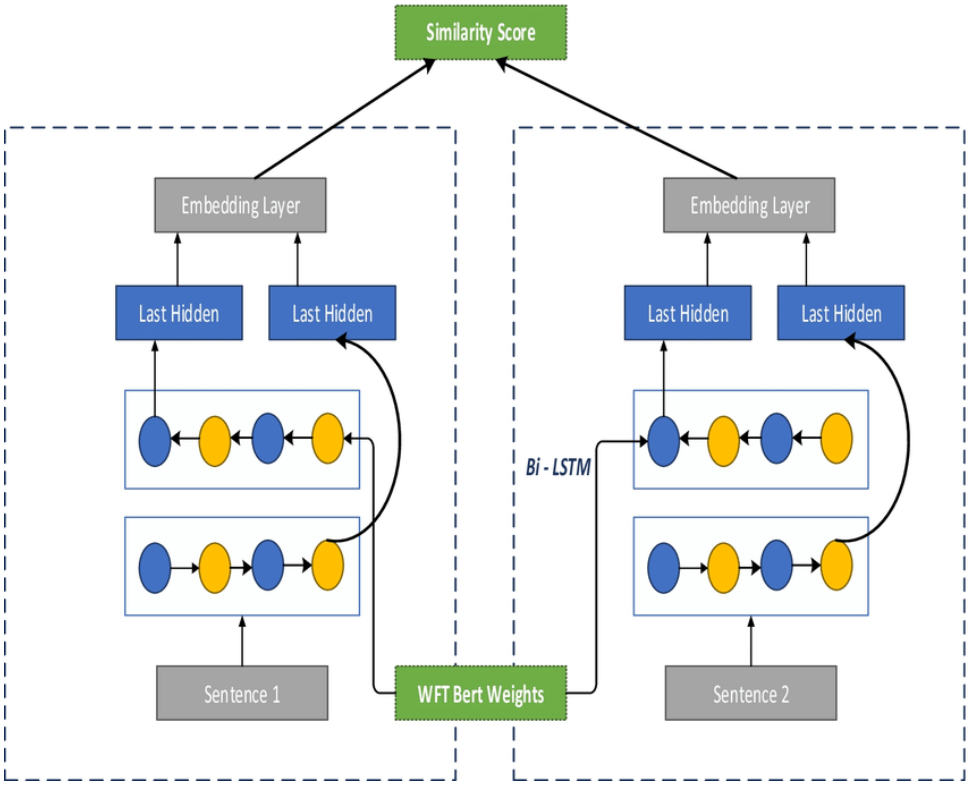


Figure 2.10: LSTM Model Architecture

- **Embedding Layer**

The initial stage in the Siamese LSTM model involves an embedding layer, whose role is to convert discrete input sequences—each represented by tokenized integers—into dense vectors of fixed size. Each token within the sequences is transformed into a 100-dimensional vector, enabling the model to capture semantic properties of words in a continuous vector space. The size of the embedding space, or the embedding dimension, is a tunable hyperparameter that affects the granularity at which the embeddings can represent word meanings.

- **Input Layers**

Two distinct input layers are constructed, one for each sequence to be compared. These layers accept sequences of fixed length, pre-processed to a standardized length through padding or truncation to ensure uniformity across the dataset. This standardization is crucial for batch processing in deep learning models.

As our example

Sequence 1 (Student's Answer): "To simulate portions of the desired final product with a quick and easy program that does a small specific job it is a way to help see what the problem is and how you may solve it in the final project."

Sequence 2 (Target Answer):"To simulate the behaviour of portions of the desired software product."

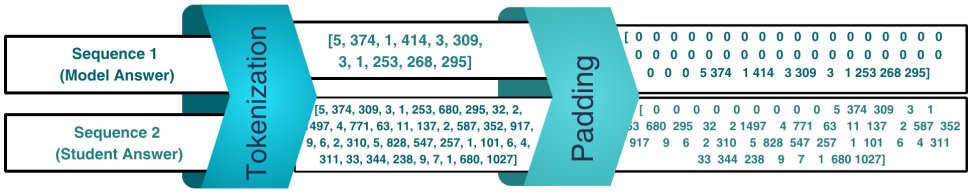


Figure 2.11: input preparation

- **Bidirectional LSTM Layer**

  Following the embedding stage, both sequences are passed through a shared Bidirectional LSTM layer. This layer enhances the model's ability to capture context by processing each sequence in both forward and reverse directions, thus gathering comprehensive temporal dynamics. The Bidirectional LSTM outputs a sequence of vectors for each input sequence, with each vector representing the state of the LSTM at a specific time step.

- **Global Average Pooling**

  To condense the variable-length outputs of the LSTM into fixed-size vectors, a Global Average Pooling layer is employed. This layer computes the average of all LSTM output vectors along the time dimension, effectively reducing each sequence to a single vector that summarizes its content.

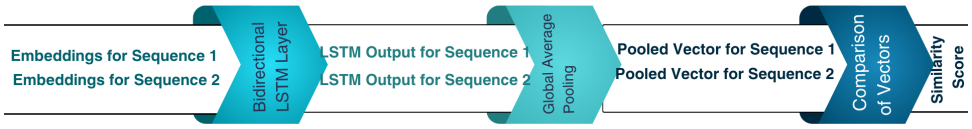


Figure 2.12: Siamese LSTM Mode

- **Siamese LSTM Model Parameters**

  - **Embedding Dimension:**we set at 100, this parameter determines the size of the dense vector space for embeddings. It directly influences how detailed the embeddings are, with higher dimensions generally capturing more nuanced word relationships.
  - **Vocabulary Size:** The size of the vocabulary for the embedding layer is set to the number of unique tokens in the training dataset (tokenizer.word-index) plus one, accounting for any out-of-vocabulary words encountered during model application.
  - **Sequence Length:** Each input sequence is standardized to a length of 50 tokens, a balance between trimming excessive length and maintaining sufficient contextual information.
  - **LSTM Units:** The Bidirectional LSTM layer utilizes 128 units, offering a robust capacity to model sequence data and capture complex dependencies.

  - **Siamese LSTM Model Parameters**

    In this model, each token is represented as a vector of 100 dimensions. The embedding dimension serves as a hyperparameter that can be adjusted based on the model's performance and the specific characteristics of the dataset. The size of the vocabulary for the embedding layer is determined by the total number of unique tokens present in the dataset, represented by tokenizer.word-index, plus one additional token to handle any out-of-vocabulary words encountered during processing.

C.ELMO Architecture We uses TensorFlow and TensorFlow Hub to leverage a pre-trained ELMO model for converting text data into embeddings. First,we imports the necessary libraries and loads the ELMO model from TensorFlow Hub.Then we define a function elmo-vectors that accepts a list of text strings, transforms them into TensorFlow tensors, retrieves embeddings using the ELMO model, and converts these embeddings back into NumPy arrays for easier manipulation. Finally, the function is applied to two columns of a DataFrame, student-answer and desired-answer, using a lambda function that processes each text entry through the ELMO model to generate corresponding embeddings, which are stored in new columns within the DataFrame. the figure 2.13 [4]show the Architecture

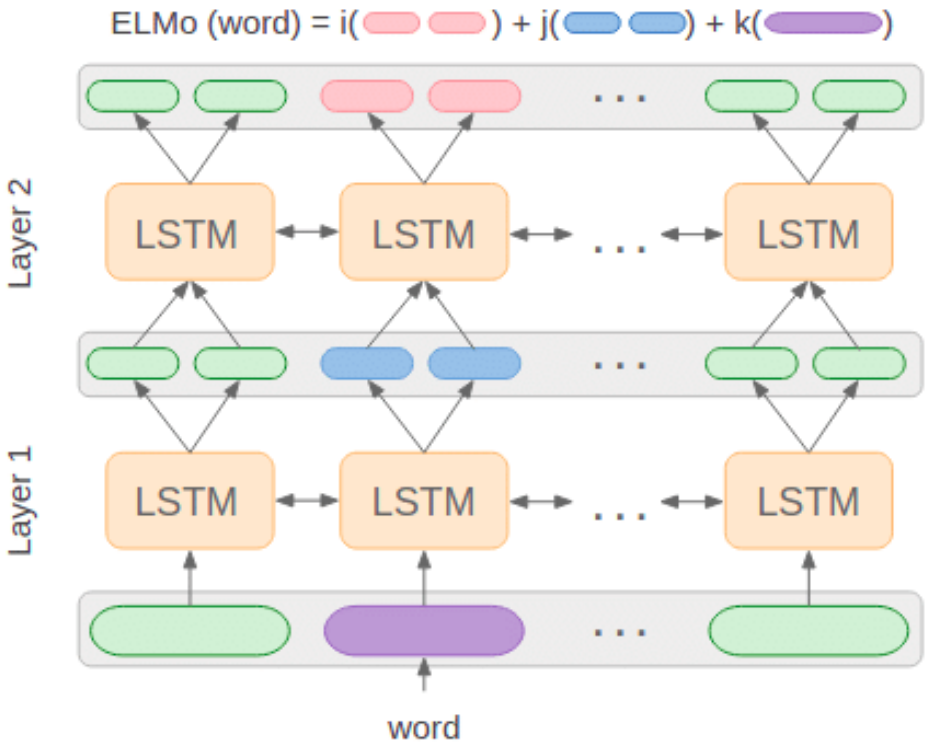


Figure 2.13: ELMO Architecture

For example
Question : Where do C++ programs begin to execute?
Desired-answer: the main function
Student-answer: main

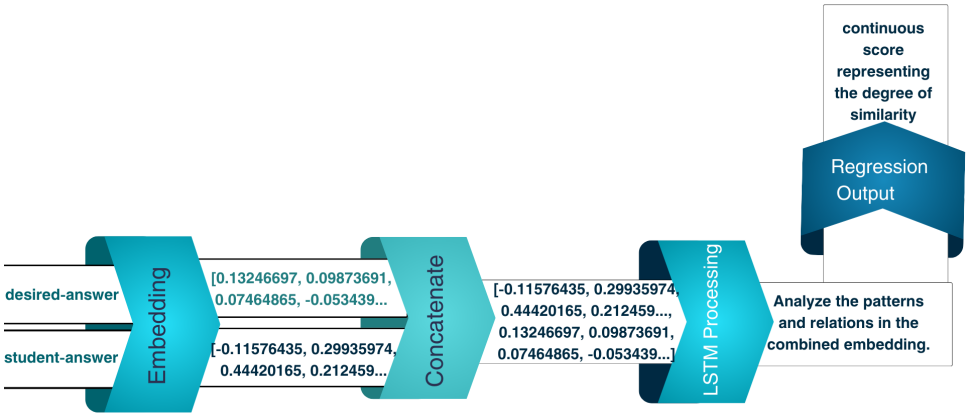


Figure 2.14: Embedding elmo

2.2.4 Grade predictor

The proposed grade prediction model utilizes a Multilayer Perceptron (MLP) designed to estimate scores within a 0 to 5 range. The architecture of the MLP is specifically tailored to complement the embedding model it accompanies:

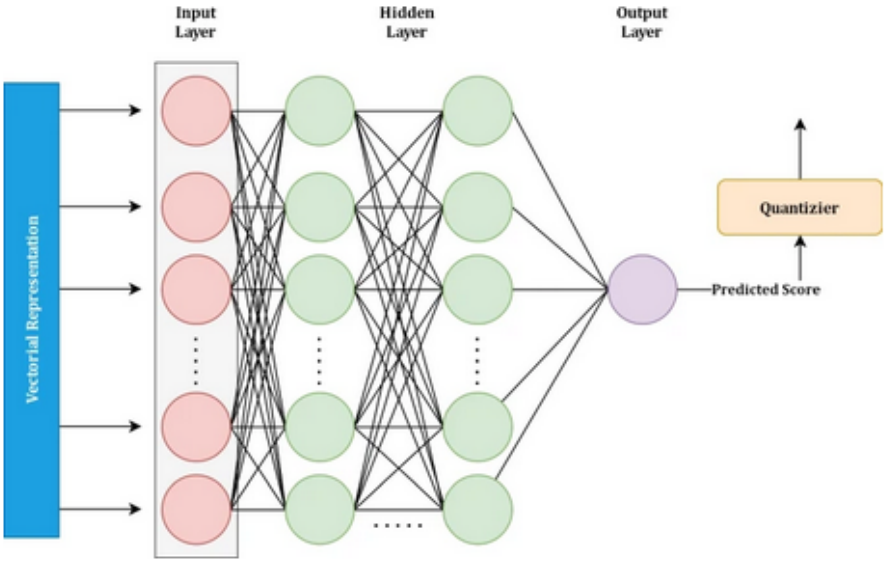


Figure 2.15: supervised-learning

We implemented an MLP component comprising two dense hidden layers, each with 768 neurons. initialized the weights using the Xavier uniform method via a custom weights-init function, targeting linear layers like torch.nn.Linear to maintain constant variance in the weights-essential for stabilizing the learning process. The first layer employs a Tanh activation function to introduce non-linearity, while the second uses a linear activation function for continuous output. A dropout rate of 0.1 after each layer helps prevent overfitting. The MLP concludes with a single neuron in the final output layer for the regression task, effectively predicting grades.

Each MLP configuration—whether paired with BERT has been meticulously fine-tuned to translate language embeddings into accurate, consistent grade predictions that reflect the quality of the student's response and the characteristics of each embedding model.

The MLP regressor processes vector representations of answered exercises, using backpropagation for training. The model's optimization focuses on minimizing the Mean Squared Error (MSE) loss function, quantifying the discrepancy between the actual expert-assigned grades and the predicted grades.

Post-processing, the predicted scores are transformed by a Quantizer we developed, which rounds each floating-point prediction to the nearest integer. This straightforward approach effectively converts continuous outputs into discrete values, crucial for scenarios requiring more interpretable predictions. we utilized a list comprehension for efficient iteration and rounding of each prediction, ensuring the model's outputs are practical and align closely with typical grading standards, thus enhancing their applicability in educational settings. I take a list of numerical predictions and rounds each number to the nearest predefined value (0.0 to 5.0 in 0.5 increments), based on a threshold of 0.25. For each prediction in the list, the function checks how close it is to these fixed points and assigns the value to the closest point if the difference is less than 0.25. The function ensures all values are effectively rounded to these specific increments, returning a list of these quantized values. However, it includes redundant conditions for values greater than 0.25 away from 0 and 5, which are logically unreachable and do not affect the output due to the comprehensive coverage of earlier conditions.

2.2.5 Evaluation

The evaluation metrics used are Pearson's correlation coefficient (r) and the Root Mean Squared Error (RMSE), both crucial in the statistical evaluation of model performance. Pearson's cor-

relation coefficient, defined as

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}}$$

measures the linear relationship between two continuous variables. The coefficient ranges from -1 to 1, where 1 or -1 indicates a perfect linear relationship, and 0 indicates no linear relationship at all. This metric is particularly effective in assessing the strength and direction of the linear relationships between predicted scores and actual expert scores.

On the other hand, RMSE, calculated by

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{T}(\hat{y}_t - y_t)^2}{T}}$$

quantifies the average magnitude of the errors between the values predicted by a model and the values actually observed. This metric is always non-negative, and a value of 0 indicates perfect accuracy. RMSE is essential for evaluating the accuracy of models in automated short answer grading, comparing the predicted scores with the actual scores provided by human experts. Both of these metrics provide comprehensive insights into different aspects of model performance, making them indispensable for regression analysis.

Additionally, Accuracy is defined as the ratio of correctly predicted instances to the total instances in the dataset, expressed as:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

Accuracy is used to determine the portion of the total number of predictions that were correct. It is particularly useful in binary and multi-class classification tasks to provide a clear measure of the model's overall classification performance.

Together, these metrics provide comprehensive insights into different aspects of model performance, making them indispensable for regression and classification analysis.

2.3 Upload the model

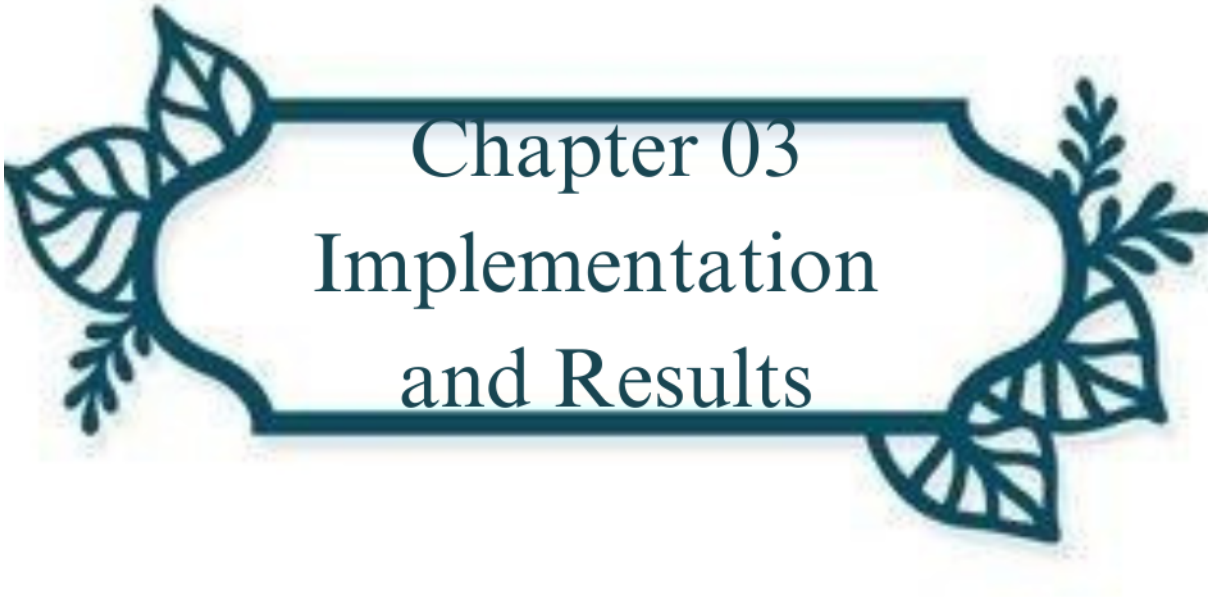
We import the onnxruntime library (aliased as ort) and use it to initialize an inference session for a model stored in the ONNX (Open Neural Network Exchange) format. The model, named "asag-model-bert-english.onnx", is loaded into an ort. InferenceSession, which prepares it for executing predictions or inferences. This approach allows you to leverage the ONNX model across various frameworks, ensuring flexibility and compatibility in deploying machine learning models that were developed in different tools or environments.

Conclusion

In conclusion, the design phase has carefully established the foundational approaches and metrics for evaluating the proposed models. The utilization of Pearson's correlation coefficient and the Root Mean Squared Error (RMSE) ensures that the models can be rigorously assessed for accuracy and effectiveness in grading.

The exploration of diverse models such as BERT,Bi-LSTM and ELMO highlights my approach to employing advanced computational techniques to improve the precision of automated grading systems. Each model has been specifically configured to optimize its performance for detailed textual analysis.

As we move from the theoretical aspects of the design to practical implementation, my focus will shift to the application of these models in real-world scenarios. The "Implementation" section will address the practical deployment of these models, tackling real-world challenges and refining the system based on iterative feedback and performance evaluations. This next step is essential for transforming theoretical designs into functional tools that meet educational demands.



Chapter 03
Implementation and Results

Chapter 3

Implementation and Results

Introduction

In this chapter, we will present and discuss the various approaches. We will begin by describing the dataset and the preprocessing techniques we applied. Following this, we will detail the data training methods used to enhance the dataset.

Next, we will outline the environment and tools employed in my model architectures, including any modifications and improvements we made. we will then cover the processes of compiling, training, and validating the models.

Finally, we will conclude with a discussion of the results, summarizing my findings and providing insights into the effectiveness of my approaches.

3.1 Environment

3.1.1 TensorFlow

TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks,TensorFlow can be used in a wide variety of programming languages, including Python, JavaScript, C++, and Java facilitating its use in a range of applications in many sectors [1].

3.1.2 Pytorch python library

PyTorch is a fully featured framework for building deep learning models, which is a type of machine learning that's commonly used in applications like image recognition and language processing. Written in Python, it's relatively easy for most machine learning developers to learn and use[2].

3.1.3 Scikit-learn python library

Scikit-learn is the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction[3].

3.1.4 Hugging Face

Hugging Face is a machine learning (ML) and data science platform and community that helps users build, deploy and train machine learning models. It provides the infrastructure to demo, run and deploy artificial intelligence (AI) in live applications. Users can also browse through

[1]https://www.tensorflow.org/
[2]https://pytorch.org/
[3]https://scikit-learn.org/stable/

models and data sets that other people have uploaded. Hugging Face is often called the GitHub of machine learning because it lets developers share and test their work openly. Hugging Face is known for its Transformers Python library, which simplifies the process of downloading and training ML models. The library gives developers an efficient way to include one of the ML models hosted on Hugging Face in their workflow and create ML pipelines [4].

3.1.5 nlpaug

nlpaug is a versatile Python library for augmenting natural language processing (NLP) datasets, designed to enhance model training and performance. It offers various augmentation techniques, including synonym replacement using WordNet, back translation for introducing variability in text, and contextual word embeddings from models like BERT to insert or replace contextually appropriate words. Additionally, nlpaug supports audio data augmentation with features like noise injection and pitch shifting. This library is particularly useful for creating more diverse training data, helping to improve model robustness and reduce overfitting, especially in scenarios with limited or imbalanced datasets[5].

3.2 Hardware used

Windows 10 operating system, Intel Core i5 processor and 8GB RAM.

The experiments were performed using a Google Colab environment with the following characteristics: 12 GB RAM and a Tesla P100-PCIE-16GB GPU.

3.3 Dataset

Mohler is the dataset provided by Rada Mihalcea. It is a computer science domain dataset. It consists of 2273 answers with 80 questions collected from 31 students of University of North Texas. The scores are graded from 0(not correct) to 5(totally correct). The grades are allocated by two evaluators. We consider their average as the gold standard score. The mean and median of the assigned gold standard scores are 4.17 and 4.50 respectively. This explains the dataset biasity towards correct answer, which may affect on the regression. [6]

The figure 3.1 show example from data:

	id	question	desired_answer	student_answer	score_me	score_other	score_avg
0	1.1	What is the role of a prototype program in pro...	To simulate the behaviour of portions of the d...	High risk problems are address in the prototyp...	4.0	3.0	3.5
1	1.1	What is the role of a prototype program in pro...	To simulate the behaviour of portions of the d...	To simulate portions of the desired final prod...	5.0	5.0	5.0
2	1.1	What is the role of a prototype program in pro...	To simulate the behaviour of portions of the d...	A prototype program simulates the behaviors of...	5.0	3.0	4.0
3	1.1	What is the role of a prototype program in pro...	To simulate the behaviour of portions of the d...	Defined in the Specification phase a prototype...	5.0	5.0	5.0
4	1.1	What is the role of a prototype program in pro...	To simulate the behaviour of portions of the d...	It is used to let the users have a first idea ...	3.0	3.0	3.0

Figure 3.1: Data's example

[4]https://huggingface.co/

[5]https://pypi.org/project/nlpaug/0.0.5/

[6]https://www.kaggle.com/datasets/abdokamr/mohler

3.4 Splitting the data

Categories	Num questions	Num answers	Num target answers
Training	1819	1819	1819
Testing	454	454	454

Table 3.1: Splitting the data

We divided our dataset based on the number of incorrect and correct answers, with one group containing entries with 0 to 3 incorrect answers and the other group containing entries with 4 to 5 correct answers.

3.5 Augmentation

We utilized the nlpaug library to augment text data for low-scoring student answers in a dataset. Initially, the script imports necessary libraries and downloads NLTK packages for word augmentation. The augment-text function leverages nlpaug to replace words in the text with their synonyms from WordNet, generating slightly altered versions of the original texts. we focused on augmenting texts with an average score of three or lower to enhance the diversity and quantity of examples for these cases. The augmented data is then formatted into a new DataFrame with updated IDs and preset low scores before being merged back into the original dataset. This strategy adds varied examples of low-scoring answers to the dataset, which is beneficial for training machine learning models to better understand and evaluate similar responses.

To balance our dataset, we augmented underrepresented data points using textual data augmentation. Specifically, we employed the nlpaug library to create variations of student answers using synonyms from the WordNet database for scores that were underrepresented. The script identifies scores with fewer instances than a target threshold (200 in this example) and calculates the necessary number of augmented versions to meet this target. For each underrepresented score, the script generates multiple existing answers, modifies them through synonym substitution, and appends these new variants to the dataset. This approach enriches the dataset, ensuring that models trained on this data are exposed to a more uniform distribution of examples across different scores. Ultimately, the outcome is a balanced dataset where each score category is equally represented, enhancing the robustness of subsequent analyses or predictive models.Then the data will be splitting like this

Categories	Num questions	Num answers	Num target answers
Training	5209	5209	5209
Testing	1303	1303	1303

Table 3.2: Splitting the data after augmentation

Figure 3.2 compares the distribution of correct and incorrect answers in our dataset before and after applying data augmentation techniques.

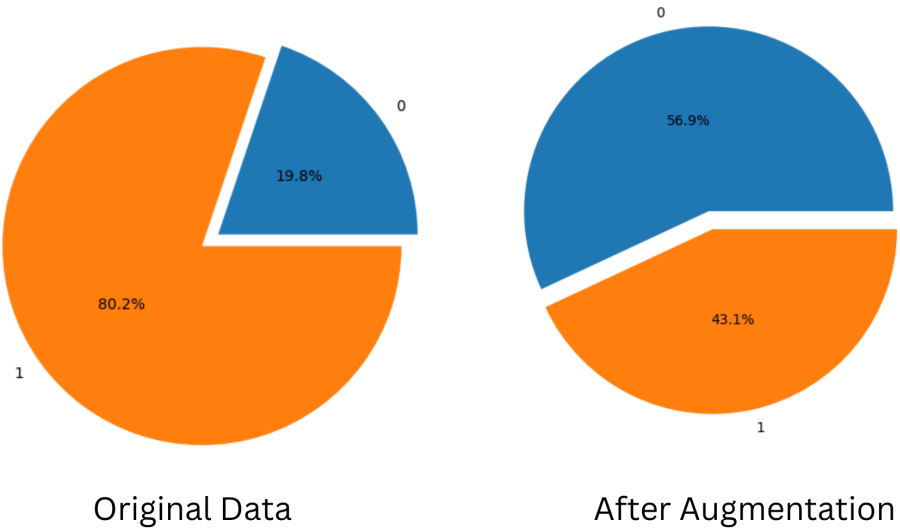


Figure 3.2: Distribution of answers by correctness

Figure 3.3 illustrates the changes in the distribution of average scores before and after data augmentation.

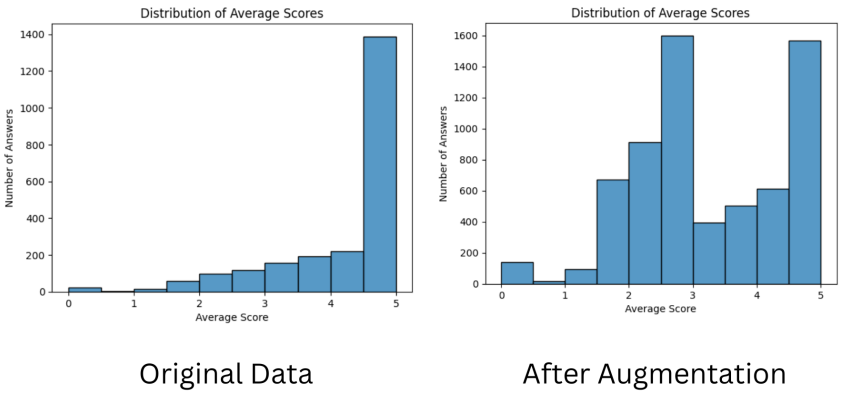


Figure 3.3: Distribution of Average Scores

To choose the best approach, a set of experiments were carried out to test the performance of the proposed alternatives. The experiments performed to test each approach are described below.

3.6 BERT Results

Our model exhibits a complex learning behavior across multiple epochs, evidenced by various performance metrics. It initially learns very quickly, as shown by the sharp decline in both training and validation losses and a swift improvement in validation accuracy. This indicates that the model's initial settings and architecture are effectively capturing the significant patterns from the dataset.However, the model's learning appears to plateau around 0.82 accuracy by epoch 12, suggesting that it has potentially reached the limits of what it can learn from the data with its current setup. the figure 3.4 show accuracy over 15 epochs

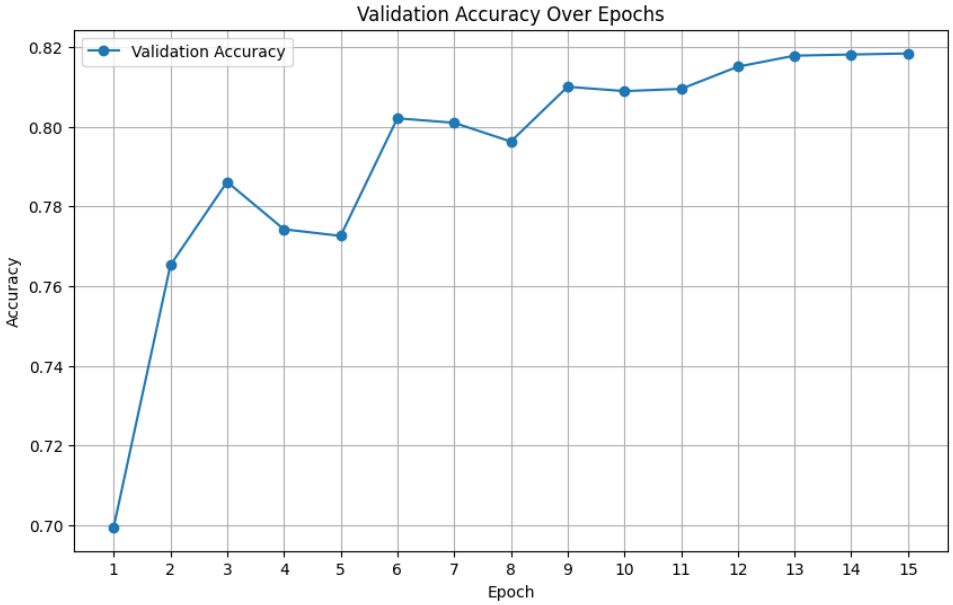


Figure 3.4: Accuracy over epochs

After augmentation my model exhibits rapid learning capabilities, evidenced by a steep increase in validation accuracy, which quickly stabilizes at a high level above 0.92. This early performance suggests that the model's initial configuration and architecture are efficiently capturing critical patterns from the dataset, especially after the application of data augmentation techniques. However, as the training progresses, the model's learning curve appears to level off, with the validation accuracy peaking around 0.94 by epoch 12 and maintaining a consistent performance thereafter. the figure 3.5 show accuracy over 25 epochs

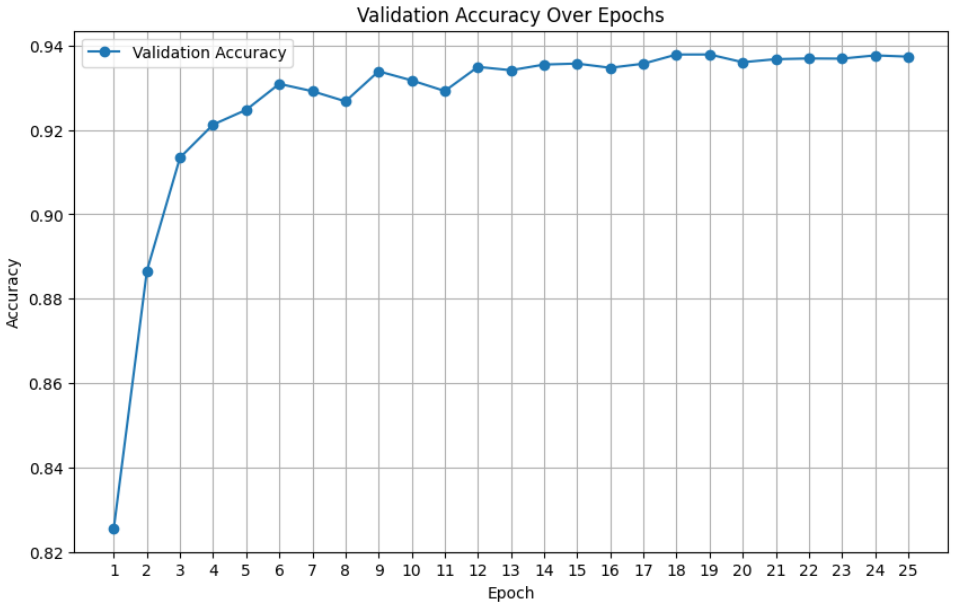


Figure 3.5: Accuracy over epochs

The consistent gap between the training and validation loss suggests that the model is fitting well to the training data without over fitting, as indicated by the lack of a substantial divergence between the two metrics.the figure 3.6 show Training validation loss over 15 epochs

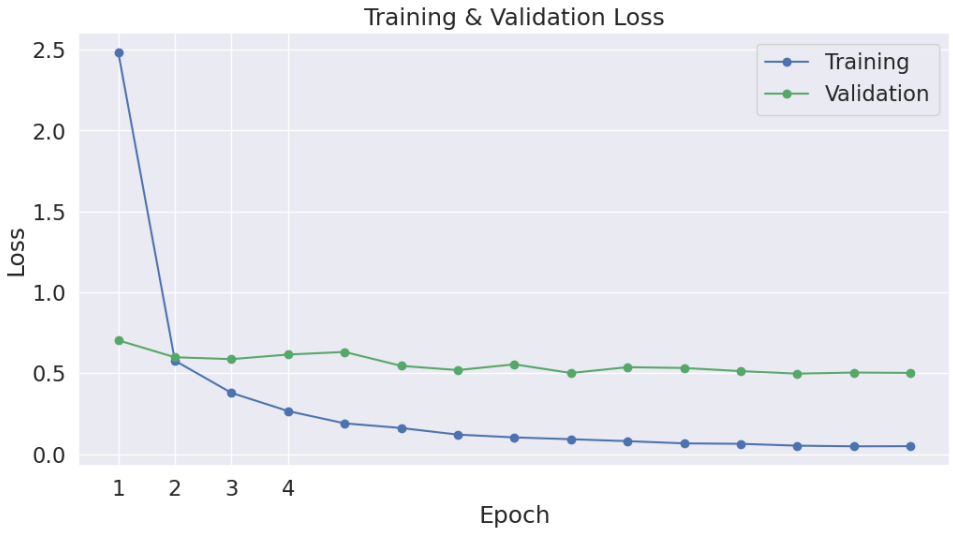


Figure 3.6: Training validation loss

After augmentation the graph indicate effective and rapid learning by the model. The sharp drop in losses at the onset followed by a gradual flattening out suggests that the model quickly adapted to the patterns in the augmented dataset, achieving a low and stable loss quite early. This behavior indicates a robust model that generalizes well across the augmented data, maintaining a consistent performance as training progresses.the figure 3.7 show Training validation loss over 25 epochs

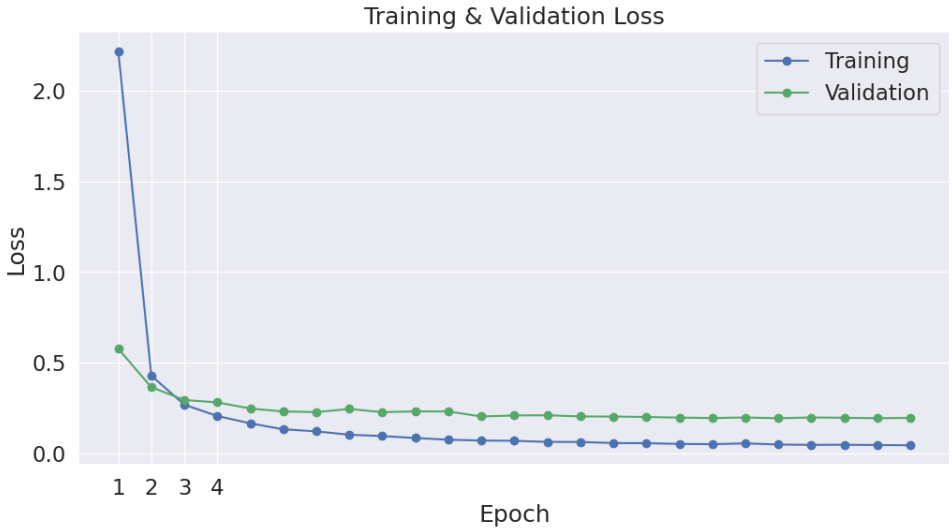


Figure 3.7: Training validation loss

The detailed table of metrics across 25 epochs reveals a consistent improvement in loss reduction coupled with varying accuracy levels. The training accuracy decreases over time, possibly indicating a shift from potential over fitting towards better generalization.

Training Loss Valid. Loss Valid. Accur. RMSE Training Time Validation Time

1	2.214257	0.577276	0.825560	0.752998	0:03:31	0:00:08
2	0.429703	0.366156	0.886600	0.595737	0:03:33	0:00:08
3	0.269608	0.295579	0.913447	0.536795	0:03:32	0:00:08
4	0.207338	0.282469	0.921285	0.522556	0:03:33	0:00:08
5	0.166875	0.248354	0.924779	0.490519	0:03:32	0:00:08
6	0.134183	0.232586	0.930946	0.476136	0:03:32	0:00:08
7	0.122778	0.228870	0.929198	0.469304	0:03:33	0:00:08
8	0.104135	0.246258	0.926790	0.489259	0:03:32	0:00:08
9	0.097023	0.228577	0.933925	0.471200	0:03:33	0:00:08
10	0.086199	0.232830	0.931762	0.475254	0:03:33	0:00:08
11	0.076844	0.232477	0.929243	0.469743	0:03:33	0:00:08
12	0.072433	0.204779	0.934967	0.434752	0:03:33	0:00:08
13	0.070749	0.210393	0.934175	0.442085	0:03:33	0:00:08
14	0.064942	0.211598	0.935524	0.439921	0:03:33	0:00:08
15	0.064403	0.204950	0.935747	0.436977	0:03:33	0:00:08
16	0.058408	0.204552	0.934753	0.436330	0:03:32	0:00:08
17	0.057845	0.201935	0.935730	0.430938	0:03:32	0:00:08
18	0.053687	0.198483	0.937899	0.433910	0:03:32	0:00:08
19	0.052129	0.196228	0.937911	0.430614	0:03:32	0:00:08
20	0.056717	0.199223	0.936070	0.425589	0:03:32	0:00:08
21	0.050323	0.194999	0.936798	0.425199	0:03:32	0:00:08
22	0.048458	0.199488	0.936964	0.428650	0:03:32	0:00:08
23	0.049190	0.197573	0.936920	0.424819	0:03:32	0:00:08
24	0.047467	0.195316	0.937698	0.423821	0:03:32	0:00:08
25	0.046017	0.196959	0.937394	0.425283	0:03:32	0:00:08

Figure 3.8: Table of metric

The table from my Automated Short Answer Grading (ASAG) system provides a clear snapshot of both manual and automated grading processes, illustrating the model's efficacy in scoring student answers relative to human graders. Notably, the predicted scores align closely with the manually computed average scores, suggesting a high degree of model accuracy in mimicking human judgment, particularly highlighted by entries. The the figure 3.9 show an example data with new column witch is predicted-score from BERT Model :

	id	question	desired_answer	student_answer	score_me	score_other	score_avg	correct	predicted	test_label
3727	1467.9	How can an array be addressed in pointer/offse...	By initializing a pointer to point to the firs...	['multi - dimensional array']	2.5	2.50	2.500	0	2.060042	2.500
860	5.3	What is the number of operations for insertion...	N (the length of the array) operations achieve...	n operations The best case scenario is when a...	5.0	5.00	5.000	1	3.720124	5.000
2094	12.4	Briefly, how does selection sort work?	It selects the minimum from an array and place...	by selecting the first object and comparing it...	1.0	3.75	2.375	0	2.431156	2.375
5930	12.9	How do you delete a node from a binary search ...	Find the node, then replace it with the leftmo...	You remove the element from the tree and move ...	5.0	3.75	4.375	1	4.548698	4.375
3476	1216.9	How many steps does it take to search a node i...	The height of the tree.	["data technology depend on the install huntin...	2.0	2.00	2.000	0	2.237236	2.000
...	...	...	...	...	...	...	...	...	...	...
289	2.3	What is the difference between a constructor a...	A constructor is called whenever an object is ...	A constructor initializes an object. A functio...	1.0	4.00	2.500	0	2.737776	2.500
2495	235.9	How does the compiler handle inline functions?	It makes a copy of the function code in every ...	['You adjudge your inline involvement in the h...	1.5	1.50	1.500	0	2.038077	1.500
3189	929.9	What are the similarities between iteration an...	They both involve repetition; they both have t...	['non firmness']	2.0	2.00	2.000	0	2.468575	2.000

Figure 3.9: Predected scores

The quantization ensures that the model's output conforms to conventional grading systems, making the results more understandable and actionable. For example, a predicted score of 5.198961 is quantize to 5. This process is critical in educational settings where precise, whole-number grading is the norm, facilitating clear communication of results and ensuring consistency in evaluation standards across different evaluators and computational models. The the figure 3.10 show an exemple of data with an other new column witch is Quantized from BERT Model :

	id	Answers	Actual Score	Predicted Score	Quantized
639	2029.9	['net explorer. GradeBook myGradeBook (); call...	2.5	2.315440	2
265	7.3	direct access to elements, and speed, linked l...	4.5	5.093356	5
260	5.4	where you only must merge sort once	3.5	3.797516	4
234	475.9	['two way of life cardinal equivalence']	1.5	1.903085	2
72	5.3	Best case is all elements sorted. n-1.	4.0	4.417992	4

Figure 3.10: Table of metric add quantize prediction

Both plots are instrumental in assessing the model's performance, indicating that while the model performs well in certain scenarios, ;,which shows actual scores versus predicted scores, we see a clear trend where predicted scores tend to cluster around whole numbers, indicating the use of quantization or rounding in the model's output. However, the spread across the y-axis for each predicted score level suggests variability in how accurately the model can predict across different cases. And a comparison of human-assigned scores and model-predicted scores across numerous test samples. In this visualization, the predicted scores are plotted alongside the actual scores given by human graders.

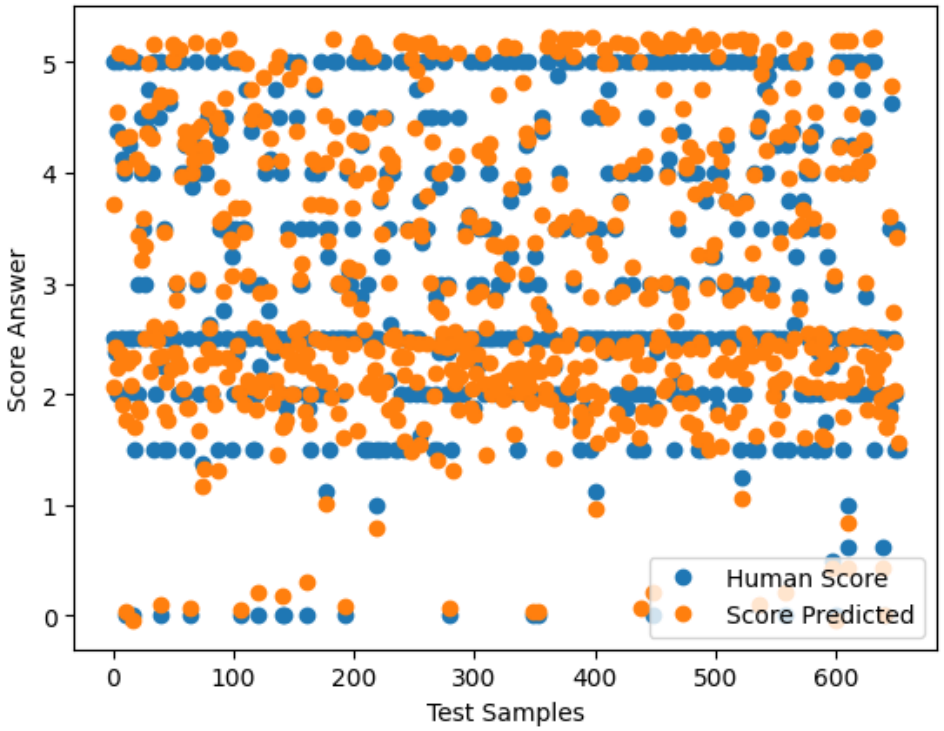


Figure 3.11: Human/Predicted Score

the differences between predicted scores (y-pred) and actual test scores (y-test) for a model, which are essential for evaluating the model's prediction accuracy and consistency. These plots

collectively highlight that my model is effective in its predictions, with most discrepancies being minor. However, the presence of some outliers where the prediction error is larger indicates potential areas for improvement.

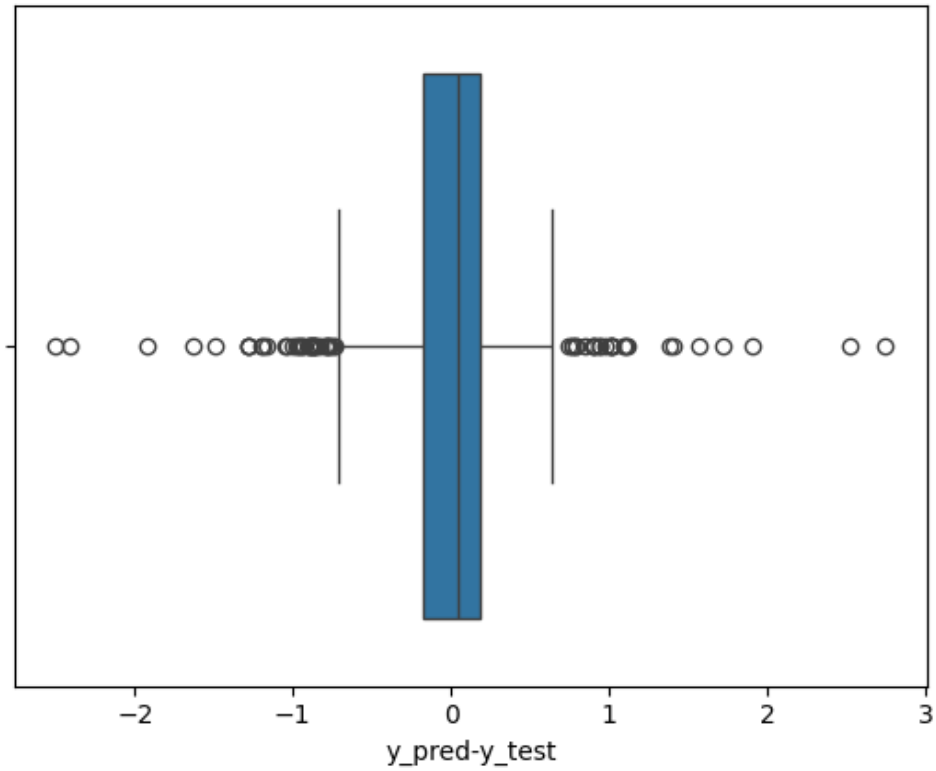


Figure 3.12: Human/Predicted Score

Comparison

The task of grading short answers automatically has garnered significant interest from researchers.These studies leverage advanced models such as SBERT, RoBERTa, BERT, to tackle the inherent challenges of semantic analysis and grading consistency. Each entry in the table provides insights into the methods employed, datasets used, and the results achieved.

Table 3.3: Summary of Research on Automatic Short Answer Grading

Article (date)	**Method**	**Data used**	**Results**
Automatic Short Answer Grading Using Deep Learning [25]	SBERT vs BERT	Short Answer Scoring V2.0	Accuracy (ACC): 0.73-0.78 Macro F1 (M-F1): 0.54-0.61 Weighted F1 (W-F1): 0.47-0.54
Multi-Relational Graph Transformer for Automatic Short Answer Grading [5]	RoBERTa Large	Mohler extended dataset	PCC 0.82 RMSE 0.056
Using BERT and XLNET for the Automatic Short Answer Grading Task [5]	BERT Base (uncased)	SciEntBank Dataset	2-way, 3-way, 5-way ACC 0.798, 0.726, 0.66 F1 W 0.792, 0.622, 0.484
Survey on Automated Short Answer Grading with Deep Learning: from Word Embeddings to Transformers [20]	BERT and Transformer Model	SciEntsBank and Mohler datasets	PCC: 0.82 MAE:0.12
Our Model using Mohler Dataset	Our BERT Model	Mohler Dataset	PCC: 0.937, RMSE: 0.2103, ACC: 0.9314

Analysis of Results

Upon comparing the performance metrics of various models with our results, several observations can be made:

- **Pearson Correlation Coefficient (PCC)**
  Our model achieved a PCC of 0.937, which is higher than the PCC achieved by RoBERTa Large (0.82) and BERT and Transformer Model (0.82). This indicates a better linear relationship between predicted and actual scores in my model.

- **Root Mean Square Error (RMSE)**
  Our model's RMSE is 0.2103, which is higher than RoBERTa Large's RMSE of 0.056. This suggests that while my model has a strong linear correlation, there is room for improvement in minimizing prediction errors.

- **Accuracy and F1 Scores**
  Our model's accuracy (0.9314) is competitive with the accuracies reported in previous studies (e.g., 0.798, 0.726, 0.66 for BERT Base).

3.7 Siamese Bidirectional LSTM Results

The results of my test on Siamese bi-LSTM shows in the table

Metric	Value
Mean Squared Error (MSE)	0.4061879444791387
Mean Absolute Error (MAE)	0.4375235932498799
Root Mean Squared Error (RMSE)	0.6373287569842888
Correlation Coefficient	0.8259318455780677

Table 3.4: Performance Metrics

Before augmentation
it left shows original continuous predictions versus actual scores, displaying wide dispersion and indicating variability in prediction accuracy. The right plot, displaying quantized predicted scores, shows a discrete categorization of predictions, which align neatly on a grid, simplifying interpretation and potentially aligning better with practical uses such as educational grading. This quantized approach reduces continuous scores to distinct groups, which could be more beneficial in settings where exact numerical precision is less critical than correct category placement

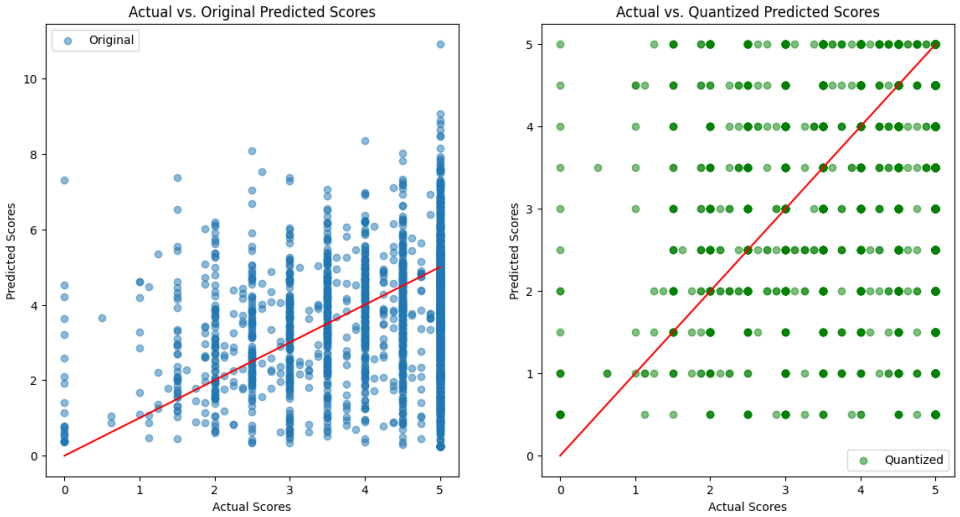


Figure 3.13: maximum length

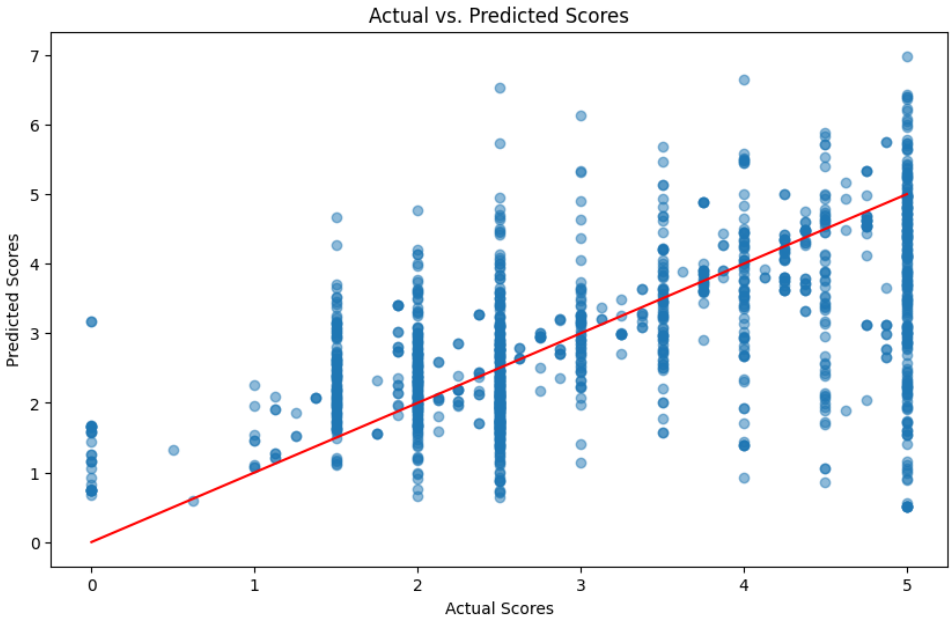


Figure 3.14: maximum length

After augmentation:

Metric	**Value**
Mean Squared Error (MSE)	0.2636383065064213
Mean Absolute Error (MAE)	0.38544143627270316
Root Mean Squared Error (RMSE)	0.513456941335319
Correlation Coefficient	0.919084180088323

Table 3.5: Updated Performance Metrics

The model shows a generally positive correlation between actual and predicted scores, indicating that it captures the underlying trend effectively. The the figure 3.15 show Score-avg Vs predicted-score :

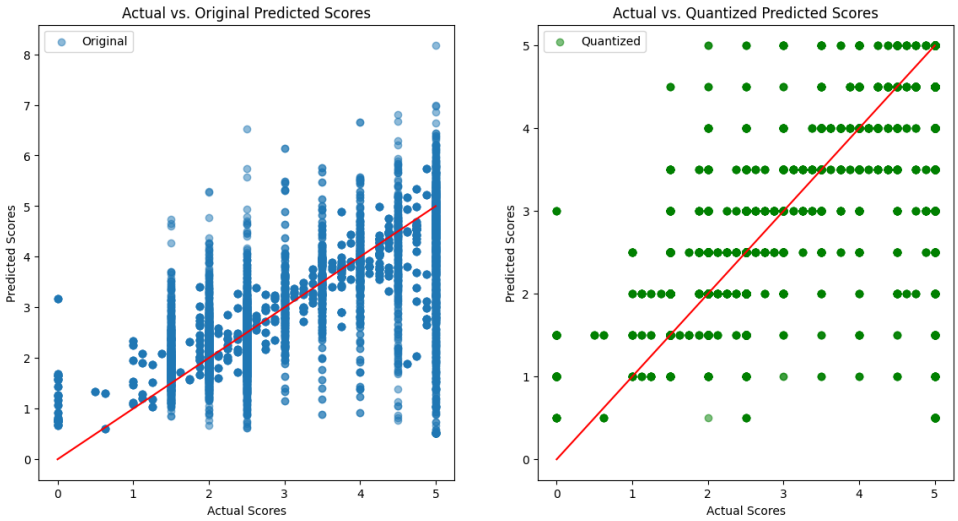


Figure 3.15: score vs predicted-score

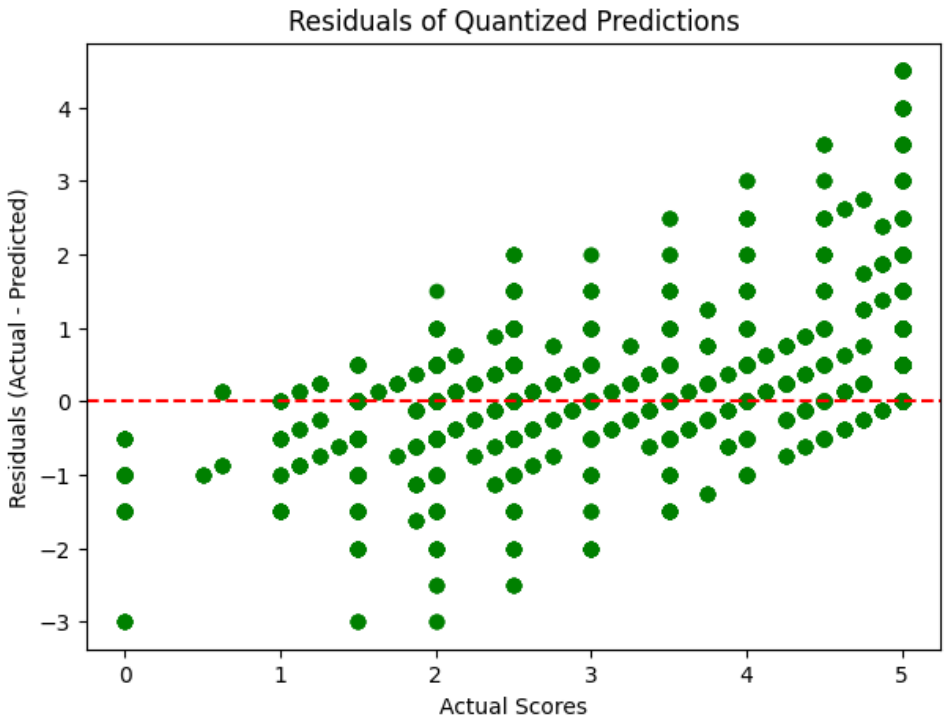


Figure 3.16: score using quantize

This table showcases a segment of our dataset used in the automated short answer grading system. along with 'predicted-score' and 'quantized-predicted-score', which are outputs from our predictive model Siamese Bidirectional LSTM. This data is pivotal for analyzing the effectiveness and accuracy of our grading system. The the figure 3.17 show example of data with new columns witch is predicted-score and Quantizered from Bi-LSTM model :

	id	question	desired_answer	student_answer	score_me	score_other	score_avg	correct	predicted_score	quantized_predicted_score
0	1.1	What is the role of a prototype program in pro...	To simulate the behaviour of portions of the d...	High risk problems are address in the prototyp...	4.0	3.0	3.5	1	2.886771	3.0
1	1.1	What is the role of a prototype program in pro...	To simulate the behaviour of portions of the d...	To simulate portions of the desired final prod...	5.0	5.0	5.0	1	5.165121	5.0
2	1.1	What is the role of a prototype program in pro...	To simulate the behaviour of portions of the d...	A prototype program simulates the behaviors of...	5.0	3.0	4.0	1	4.711360	4.5
3	1.1	What is the role of a prototype program in pro...	To simulate the behaviour of portions of the d...	Defined in the Specification phase a prototype...	5.0	5.0	5.0	1	3.976392	4.0
4	1.1	What is the role of a prototype program in pro...	To simulate the behaviour of portions of the d...	It is used to let the users have a first idea ...	3.0	3.0	3.0	0	2.263207	2.5

Figure 3.17: score using quantize

3.7.1 Comparison

In the table 3.6 we compare our model with previews studies with Bi-LSTM

Table 3.6: Summary of Model Evaluations on the Mohler Dataset

Architecture Used	**PCC**	**RMSE**	**MAE**	**Dataset Used**
Siamese Bidirectional LSTM Architecture	0.651	0.930	0.668	Mohler dataset
Siamese Bidirectional LSTM Architecture with Feature Engineering	0.655	0.889	0.618	Mohler dataset
Simple LSTM Architecture	0.381	0.982	0.959	Mohler dataset
LSTM Architecture with Feature Engineering	0.523	0.901	0.904	Mohler dataset
Our Model	0.918	0.485	0.354	Mohler dataset

Analysis of Results

Performance Metrics :
The Siamese Bidirectional LSTM Architecture with Feature Engineering achieved the best results in my study, with a PCC of 0.655, RMSE of 0.889, and MAE of 0.618. The Simple LSTM Architecture performed the worst in my study, with a PCC of 0.381, RMSE of 0.982, and MAE of 0.959.

Training and Test Errors :
The training loss and test MSE for Our models were both 1.3920, indicating a consistent performance between training and testing phases. The MAE on the test set was 0.8466, which is relatively high but shows room for improvement. While Our models perform competitively, there is still room for improvement, particularly in reducing the error metrics further. This analysis underscores the need for continuous experimentation with different model architectures and feature engineering techniques to achieve optimal performance in automated short answer grading systems.

3.8 ELMO Results

The results of my test on ELMO shows in the table

Metric	Value
Mean Squared Error (MSE)	0.24943214092610652
Mean Absolute Error (MAE)	0.34550803878179487
Root Mean Squared Error (RMSE)	0.49943181809542986
Correlation Coefficient	0.9233845787005944
Custom Accuracy (tolerance ±0.5)	0.772831926323868

Table 3.7: Performance Metrics

This table showcases a segment of our dataset used in the automated short answer grading system. along with 'predicted-score' and 'quantized-predicted-score', which are outputs from our predictive model Elmo with LSTM Layers. This data is pivotal for analyzing the effectiveness and accuracy of our grading system.

The figure 3.18 shows example of data with new columns witch is predicted-score and Quantizered from Elmo model :

	id	question	desired_answer	student_answer	score_me	score_other	score_avg	correct	predicted_score
1263	8.1	What is a stack?	a data structure that can store elements which...	a stack is similar to an array but does not al...	5.0	5.00	5.000	1	4.969862
3951	1691.9	How are objects initialized when they are crea...	by using constructors	object comprise place asunder a space in compu...	2.5	2.50	2.500	0	1.962157
6199	11.7	What is the difference between an array that i...	the static arrays are intialized only once whe...	a static array has a predetermined size and th...	2.0	3.75	2.875	0	2.915623
6080	11.2	What are the access-specifiers that can be use...	private and public	public private and protected	5.0	4.75	4.875	1	4.770035
96	1.4	Where do C++ programs begin to execute?	at the main function	main	5.0	5.00	5.000	1	4.997571

Figure 3.18: predicted score

The plot shows a stable training loss and a higher, relatively flat validation loss across epochs, however the model learns the training data well but generalizes poorly to new data.

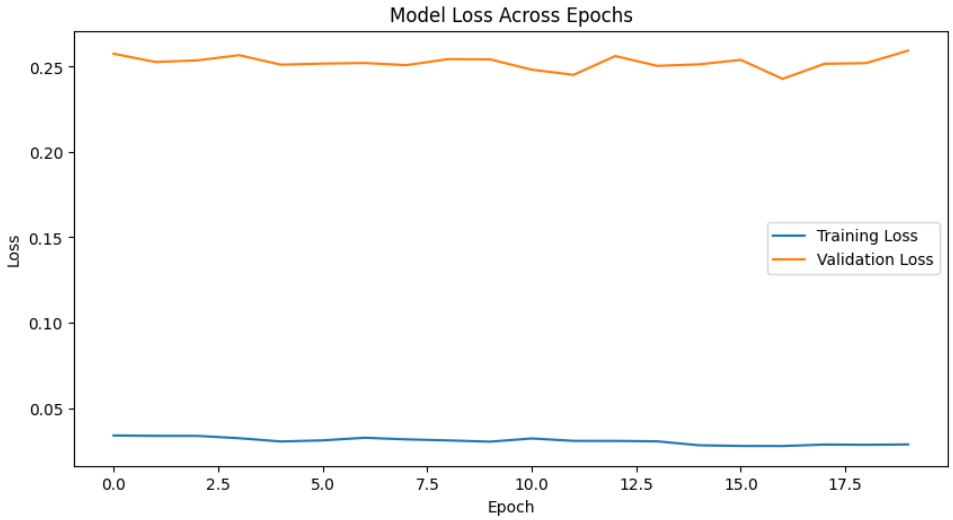


Figure 3.19: predicted score

Comparison

In this table 3.8 we compare our model with previews studies with ELMO

Table 3.8: Summary of Evaluation Metrics for Different Models

Model	CC	RMSE	Accuracy
Elmo V2.0	0.390	0.896	-
Our work	0.92	0.4988	0.77

Analyse results

In Table 3.8, the evaluation metrics of two models, Elmo V2.0 and "Our work," are compared using Correlation Coefficient (CC), Root Mean Squared Error (RMSE), and Accuracy. The results highlight the significant improvements achieved by "Our work" over the Elmo V2.0 model. Specifically, "Our work" exhibits a high CC of 0.92, indicating a strong linear relationship between the predicted and true values, and a substantially lower RMSE of 0.4988, demonstrating improved precision and reduced prediction errors. Furthermore, the accuracy of "Our work" stands at 0.77, which signifies a robust performance in terms of correctly predicting the outcomes. In contrast, Elmo V2.0 shows a much lower CC of 0.390, suggesting weaker predictive reliability, and a higher RMSE of 0.896, indicating greater deviation from the actual values. The absence of an accuracy measure for Elmo V2.0 further limits its evaluative comparison. Overall, the metrics clearly indicate that "Our work" offers a superior predictive performance and reliability compared to the Elmo V2.0 model, showcasing the advancements and effectiveness of the proposed model in the given context.

3.8.1 Comparison with other works

Table 3.9: Summary of ASAG System Performance Using Deep Learning Models

Method	Dataset Used	Accuracy (ACC)	Macro F1 (M-F1)	Weighted F1 (W-F1)
Softmax vs Cosine	Short Answer Scoring V2.0	0.73-0.77	0.54-0.56	0.46-0.48
Shorter vs Longer Answers	Short Answer Scoring V2.0	0.67-0.81	0.49-0.71	0.44-0.63

Table 3.10: Summary of Paragraph Embedding Methods and Performance Metrics

Method	**Model Dimension**	**PCC**	**RMSE**	**Dataset Used**
Sum of Pre-trained Word Vector Model				
GloVe	300	0.507	0.838	Short Answer Grading V2.0
Google word2vec	300	0.532	0.821	Short Answer Grading V2.0
FastText	300	0.519	0.831	Short Answer Grading V2.0
Training of Paragraph Vectors Model				
doc2vec	300	0.569	0.797	Short Answer Grading V2.0
InferSent with GloVe Word Vectors	4096	0.506	0.843	Short Answer Grading V2.0
InferSent with Fasttext Word Vectors	4096	0.4597	0.862	Short Answer Grading V2.0
Skip Thoughts	2100	0.468	0.861	Short Answer Grading V2.0

3.9 Analysis and discussion

In Table 3.11, the performance metrics of three NLP models—Elmo, biLSTM, and BERT—are compared using Pearson Correlation Coefficient (PCC), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Accuracy (Accu) percentage. Among these models, BERT demonstrates the highest PCC of 0.937, indicating the strongest correlation with the true values, and also achieves the lowest RMSE of 0.2103, suggesting it has the smallest average prediction error. The absence of MAE for BERT precludes direct comparison in this metric, but its superior performance is reflected in the highest accuracy of 93.14. Elmo shows a commendable performance with a PCC of 0.923 and an RMSE of 0.499, but it is slightly less accurate at 75.2. The biLSTM model, while performing robustly with a PCC of 0.918 and an RMSE of 0.485, outperforms Elmo in accuracy at 85.87, suggesting it balances well between correlation and error metrics. This comparative analysis underscores BERT's effectiveness in NLP tasks, showcasing its advanced capability to capture semantic nuances, followed by the solid performance of biLSTM, and finally, Elmo, which, while proficient, does not quite match the efficacy of the other two models in this evaluation.

Table 3.11: Comparative Performance Metrics of NLP Models

Model	PCC	RMSE	MAE	Accu (%)
Elmo	0.923	0.499	0.345	75.2
biLSTM	0.918	0.485	0.354	85.87
BERT	0.937	0.2103	-	93.14

Conclusion

In this chapter, we have implemented various approaches and conducted a comparative analysis of sentence embedding models—Elmo, biLSTM, and BERT—based on their performance metrics. Additionally, we have compared these models using augmented data to assess the impact of data augmentation on their effectiveness. In conclusion, while the BERT model provides the most accurate predictions as indicated by the highest PCC (0.937), lowest RMSE (0.2103), and highest accuracy (93.14), Elmo and biLSTM also show competitive performance. Elmo has a higher PCC (0.923) than biLSTM (0.918) and a slightly better RMSE (0.499 vs. 0.485), while biLSTM has a slightly better MAE (0.354 vs. 0.345) and significantly higher accuracy (85.87 vs. 75.2) compared to Elmo. These results indicate that BERT offers a robust balance between prediction accuracy and alignment with actual data trends, making it potentially more suitable for applications that demand high reliability in trend prediction. Additionally, the results with augmented data showed significant improvement in model performance. Moving forward, each model's unique strengths and weaknesses, along with the benefits of data augmentation, should be considered in light of the specific requirements of the intended application.

General Conclusion

Conclusion and Future Work

This study aimed to develop and validate an Automated Short Answer Grading (ASAG) system, specifically tailored for the English language. Utilizing advanced techniques in deep learning and transfer learning, we structured the ASAG system to include steps such as data splitting, sentence embedding with different models trying to find better results , grade prediction, and a comprehensive evaluation process. Our evaluation utilized critical metrics, Pearson's correlation coefficient (PCC), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Accuracy to assess model performance on the augmented Mohler dataset.

The findings from this study highlighted distinct performance characteristics among the tested models. The BERT model demonstrated the highest performance with a PCC of 0.937, indicating the strongest ability to capture trends and variability in the data. It also achieved the lowest RMSE of 0.2103, suggesting minimal prediction errors, and the highest accuracy of 93.14, making it highly effective for precise prediction tasks. In contrast, the Elmo model showed a slightly lower PCC of 0.923 and an RMSE of 0.499, with an accuracy of 75.2, indicating reliable but less optimal performance compared to BERT. The biLSTM model, while robust with a PCC of 0.918 and RMSE of 0.485, displayed a higher MAE of 0.354 and an accuracy of 85.87, suggesting that it requires further tuning to enhance its performance.

Additionally, the results with augmented data showed significant improvement in model performance across all metrics, demonstrating the positive impact of data augmentation on enhancing the effectiveness of the ASAG system.

Looking forward to future developments, there is a strong intention to integrate the BERT model into the univ-Model platform currently used by our university. This integration aims to enhance the effectiveness of our educational assessments, allowing for more nuanced and adaptive evaluations that can significantly benefit both educators and students. Improving the capability to accurately measure and understand the similarities between student answers and target answers is crucial. Such advancements will not only provide more detailed and precise feedback but are also essential for progressing the field of automated educational assessments. This area of study demands a nuanced understanding and adaptability in response evaluation, which are increasingly critical as educational technologies evolve.

Bibliography

[1] A comprehensive guide to deep learning: Concepts, methods, and applications, 2024.

[2] Harnessing supervised learning for..., 2024.

[3] Intelligence artificielle: un bouleversement plus important qu'internet, 2024.

[4] Manel Affi and Chiraz Latiri. Be-blc: Bert-elmo-based deep neural network architecture for english named entity recognition task. *Procedia Computer Science*, 192:168–181, 2021.

[5] Rajat Agarwal, Varun Khurana, Karish Grover, Mukesh Mohania, and Vikram Goyal. Multi-relational graph transformer for automatic short answer grading. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2001–2012, 2022.

[6] Abbirah Ahmed, Arash Joorabchi, and Martin J Hayes. On deep learning approaches to automated assessment: Strategies for short answer grading. *CSEDU (2)*, pages 85–94, 2022.

[7] Linda Allal and Lucie M. Lopez. Formative assessment of learning: a review of publications in french. In Janet Looney, editor, *Formative assessment: improving learning in secondary classrooms*, pages 241–264. Organisation for Economic Co-operation and Development, Paris, France, 2005.

[8] Zaira Hassan Amur, Yew Kwang Hooi, and Gul Muhammad Soomro. Automatic short answer grading (asag) using attention-based deep learning model. In *2022 International Conference on Digital Transformation and Intelligence (ICDI)*, pages 1–7. IEEE, 2022.

[9] Taiwo Oladipupo Ayodele. Machine learning overview. *New Advances in Machine Learning*, 2(9-18):16, 2010.

[10] Taiwo Oladipupo Ayodele. Types of machine learning algorithms. *New advances in machine learning*, 3(19-48):5–1, 2010.

[11] Meltem Huri Baturay. An overview of the world of moocs. *Procedia-Social and Behavioral Sciences*, 174:427–433, 2015.

[12] Steven Burrows, Irena Koprinska, and Kalina Yacef. Automated short answer grading: A review of the literature. *Journal of Educational Technology & Society*, 17(4):273–294, 2014.

[13] Jean-Pierre Cuq and Isabelle Gruca. *Cours de didactique du français langue étrangère et seconde*. PUG, Grenoble, 2003.

[14] Gilbert De Landsheere. *Evaluation continue et examens. Précis de docimologie*. Edition LABOR, 6 edition, 2000. Revue et augmentée.

[15] AuthorFirstName Demir. Understanding transfer learning: Unlocking the power of pre-trained models. `https://blog.demir.io/understanding-transfer-learning-unlocking-the-power-of-pre-trained-models-885e4775d8`, 2023. Accessed: 2024-06-17.

[16] Niklas Donges. What is transfer learning? exploring the popular deep learning approach. *Built in*, 2019.

[17] Sasi Kiran Gaddipati, Deebul Nair, and Paul G Plöger. Comparative evaluation of pretrained transfer learning models on automatic short answer grading. *arXiv preprint arXiv:2009.01303*, 2020.

[18] Hadi Abdi Ghavidel, Amal Zouaq, and Michel C Desmarais. Using bert and xlnet for the automatic short answer grading task. In *CSEDU (1)*, pages 58–67, 2020.

[19] Palash Goyal, Sumit Pandey, and Karan Jain. Deep learning for natural language processing. *New York: Apress*, 2018.

[20] Stefan Haller, Adina Aldea, Christin Seifert, and Nicola Strisciuglio. Survey on automated short answer grading with deep learning: from word embeddings to transformers. *arXiv preprint arXiv:2204.03503*, 2022.

[21] Naamah Hussein Hameed and Ahmed T Sadiq. Automatic short answer grading system based on semantic networks and support vector machine. *Iraqi Journal of Science*, pages 6025–6040, 2023.

[22] Alon Jacovi, Oren Sar Shalom, and Yoav Goldberg. Understanding convolutional neural networks for text classification. *arXiv preprint arXiv:1809.08037*, 2018.

[23] Abdulkadir Kara, Eda Saka Şimşek, and Serkan Yıldırım. The unveiling the landscape: Studies on automated short answer evaluation. *Asian Journal of Distance Education*, 19(1), 2024.

[24] Shanshan Liu, Xin Zhang, Sheng Zhang, Hui Wang, and Weiming Zhang. Neural machine reading comprehension: Methods and trends. *Applied Sciences*, 9(18):3698, 2019.

[25] Jinzhu Luo. *Automatic short answer grading using deep learning*. Illinois State University, 2021.

[26] Ivan D Mardini G, Christian G Quintero M, César A Viloria N, Winston S Percybrooks B, Heydy S Robles N, and Karen Villalba R. A deep-learning-based grading system (asag) for reading comprehension assessment by using aphorisms as open-answer-questions. *Education and Information Technologies*, 29(4):4565–4590, 2024.

[27] Dastan Maulud and Adnan M Abdulazeez. A review on linear regression comprehensive in machine learning. *Journal of Applied Science and Technology Trends*, 1(2):140–147, 2020.

[28] Suraj Jung Pandey. *Modelling Alignment and Key Information for Automatic Grading*. Open University (United Kingdom), 2022.

[29] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018.

[30] Quantum Books. The rise of massive open online courses (moocs) in higher education, 2021. [Online; accessed 17-June-2024].

[31] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[32] Ivan Vasilev, Daniel Slater, Gianmario Spacagna, Peter Roelants, and Valentino Zocca. *Python Deep Learning: Exploring deep learning techniques and neural network architectures with PyTorch, Keras, and TensorFlow*. Packt Publishing Ltd, 2019.

[33] D Viji and S Revathy. A hybrid approach of weighted fine-tuned bert extraction with deep siamese bi–lstm model for semantic text similarity identification. *Multimedia tools and applications*, 81(5):6131–6157, 2022.

[34] Daniel Wilianto and Abba Suganda Girsang. Automatic short answer grading on high school's e-learning using semantic similarity methods. *TEM Journal*, 12(1), 2023.

[35] Ryan Thomas Williams. An overview of moocs and blended learning: Integrating mooc technologies into traditional classes. *IETE Journal of Education*, pages 1–8, 2024.

[36] Changchang Zeng, Shaobo Li, Qin Li, Jie Hu, and Jianjun Hu. A survey on machine reading comprehension—tasks, evaluation metrics and benchmark datasets. *Applied Sciences*, 10(21):7640, 2020.

[37] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.