LEARNING CONTENT CURATION AND ENRICHMENT

A Dissertation
Presented to
The Academic Faculty

By

Venktesh V, PhD19016

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
Indraprastha Institute of Information Technology
Department of Computer Science and Engineering

Indraprastha Institute of Information Technology

March 2023

THESIS CERTIFICATE

This is to certify that the thesis titled **Learning Content Curation and Enrichment submitted to IIIT-D**, submitted by **Venktesh V**, to the Indraprastha Institute of Information Technology, Delhi, for the award of the degree of **Doctor of Philosophy**, is a bonafide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Mukesh Mohania	**Dr. Vikram Goyal**
Thesis Supervisor	Thesis Supervisor
Professor	Professor
Dept. of Computer Science and Engineering	Dept. of Computer Science and Engineering
IIIT Delhi, 110020	IIIT Delhi, 110020

Life is not easy for any of us. But what of that? We must have perseverance and above all confidence in ourselves. We must believe that we are gifted for something and that this thing must be attained

Marie Curie

ACKNOWLEDGMENTS

I have always been fascinated by the ability to solve important problems computationally, and this encouraged me to pursue a PhD in CSE. My thesis “Towards Personalized Education Through Content Analytics and Enrichment” was carried out in the Applied Data Science (ADS) lab. I would like to take the opportunity to thank the amazing people who helped me on this journey. I would first like to thank my advisors, Dr. Mukesh Mohania and Dr. Vikram Goyal. Their constant support and guidance helped spark new ideas and stay on course. They were very approachable and helped me shape my research vision early on, enabling me to work on applications that have real-world impact. Next, I would like to thank our partner company Extramarks and SERB-FICCI, for their generous funding for continuing my research. **Extramarks** also provided us with valuable data needed for research, and I would like to thank my mentor Ritvik and data scientist, Deep for enabling me to contribute to critical functionalities in the Edtech domain. I would also take this opportunity to thank my Mother and Father who supported me and encouraged me to pursue my doctoral studies. Their support during the pandemic and beyond was more than what I could ask for. I would also thank my peers at IIIT-D, who helped build a supportive environment conducive for brainstorming. In particular, I would like to express my heartfelt gratitude to Tharrmashastha S A P V for encouraging me to pursue more fundamental ideas and always lending an ear to my ideas and rants. His math skills helped me understand fundamental work in theoretical ML. He has been an invaluable mentor, personally and professionally. Finally, I would like to thank all Faculty members, thesis review committee members, and my yearly committee members. Their feedback has enabled me to improve my work. I also express gratitude to the administrative professionals at IIIT-D for always being approachable and friendly.

Venktesh V

V. Venktesh

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

AKE Automatic Keyphrase Extraction 18

BERT Bidirectional Encoder Representations from Transformers 13

LDA Latent Dirichlet Allocation 22

LLMs Large Language Models 6

MRR Mean Reciprocal Rank 44

MWP Math Word Problems 5, 7

PAN Pseudo Adversarial Noising 96

PAP Prompt Aware Paraphraser 95

ParaQD Paraphrase Quality Detector 93

SSMBA Self-supervised Manifold Based Augmenttaion 29

STEAD STructurE Aware Denoiser 95

SVM Support Vector Machines 11

TPR Topical Page Rank 22

UDA Unsupervised Data Augmentation 29

USE Universal Sentence Encoder 18

VAE Variational Auto Encoders 29

SUMMARY

Education in traditional classroom settings was restricted to static content in textbooks. It also assumes all the learners have a similar pace of learning. Online learning platforms have shifted the paradigm and have made *“learning from anywhere”* possible. They have also enabled to scale learning to millions of users across demographics. Such learning platforms curate content from multiple sources. They also have dedicated academicians to aid in the creation and curation of learning content like videos, lecture transcripts, assessments etc. For sake of simplicity, we refer to the various categories of content as *learning content*. Manual creation of content is cumbersome. Additionally, when on-boarding content from other sources, they have to be pre-processed to follow the organization standard of the learning management system. For catering to the needs of different stages of learners, such platforms also have to link to related content to facilitate the effective distribution of knowledge to learners.

In order to deliver learning content at scale to the learners and cater to the individual needs of the learners, the content in such platforms can no longer be static and must adapt according to the interaction of the individual learners with the system. This involves onboarding new content from other sources, organizing them for ease of access, and enrichment of existing content to generate diverse content for the learners. In our work, we build content curation and enrichment tools to assist the academicians. We achieve this by proposing novel tasks and also relating tasks to existing work in literature.

We first look at the problem of organizing content according to a standardized hierarchical learning taxonomy of form (subject - chapter - topic - sub-topic) to aid in applications like faceted search. Effective organization of learning content is a prerequisite for the recommendation of appropriate learning content. However, the label space of hierarchical learning taxonomy is large and has a heavy class imbalance at the topic level of the taxonomy. To tackle this, we propose a novel reformulation of the classic HMLTC (Hierarchical

Multi-Label Text Classification) task to a dense retrieval task. We further augment this approach with an efficient cross-attention mechanism with theoretical bounds to induce label-aware learning content representations.

Second, we demonstrate that the content can be further tagged with fine-grained academic concepts to facilitate the linkage of related content and granular content recommendations. To accomplish this, we map it to the tasks of unsupervised keyphrase extraction and set expansion. We propose a novel random walk based approach with new measures leveraging contextualized and topical representations of candidate phrases and content. The fine-grained concepts extracted aid in indexing content and enriching existing learning taxonomy. We further enrich the extracted concepts with additional concepts from Knowledge Bases (KBs) like Wikipedia to aid in linking related content.

Since e-learning platforms have diverse learners who learn at different paces, the assessments conducted should also be adaptive. The first step in adaptive assessments is the organization of content according to well-established difficulty levels and pedagogical cognitive taxonomy. We adopt the Bloom's taxonomy and related difficulty levels and propose a tool for automated categorization of content to the appropriate levels. We propose a multi-task learning system to automatically tag the difficulty level of questions while simultaneously predicting the Bloom's taxonomy levels. We propose a novel interactive attention mechanism that leverages the affinity between the two tasks. Tagging questions with difficulty levels aids in adapting questions according to learners abilities in automated assessments. The difficulty adapted questions would aid a learner in understanding the skill gap and concepts to focus on while learning. Further, we propose a framework for the novel task of paraphrasing Math Word Problems (MWP) to enrich the existing questions' repository. This renders a diverse array of questions for academicians to choose from when formulating assessments. It also facilitates adaptive assessments for learners to enhance their learning experience.

In summary, the core contributions of this work are: a) pipelines to aid academicians

in the automated curation of content at scale and b) data enrichment pipelines to provide diverse learning content to the users of the learning platform.

CHAPTER 1
INTRODUCTION

1.1 Motivation

E-learning platforms have become an integral part of education in current times. They enable distance education and accessibility of resources across demographics [1]. The learning materials available in such platforms are diverse, voluminous and are no longer limited to traditional textbooks and traditional classroom resources. The learning content in such platforms comprises multiple modalities like text, audio, and video. Due to the abundance of such diverse learning resources, it becomes difficult for the user to navigate to the relevant content. To enable ease of access to the resources, the learning content in such platforms are organized in a standardized format. The task of **content curation**, which refers to gathering learning materials from diverse sources and organizing them is performed manually by academicians [2]. The task of manual curation from such diverse data sources is cumbersome due to the large volume of resources and makes it difficult to scale the platform to deliver content to a large number of users. Additionally, the educational resources from the Web and other sources are structured according to a different format or unstructured. To ensure learners from different backgrounds are able to access the right content, they have to be organized according to a standard learning taxonomy. The learning taxonomy is usually hierarchical and comprises multiple facets like subject, topics helping in faceted search [3]. To reduce the effort involved in manual data curation, there is a need for tools that can assist academicians to onboard learning resources at scale from varied data sources.

The users of e-learning platforms are diverse and have different information needs. When accessing educational content, a learner may also be interested in accessing related content to form a better understanding of the underlying concept. For instance, to compute the density of an object immersed in water, a learner must learn about "Archimedes Principle." While the contents in learning platforms are organized according to a standardized learning taxonomy, much fine-grained meta-data like concepts are needed for recommending relevant resources. Hence, the learning resources in such platforms cannot exist in isolation and must be automatically enriched with meta-data to enable linkage to data from other data sources. The need for meta-data augmentation for learning resources termed as **content enrichment** is further supported in the recent study on the effect of learning technologies-assisted content delivery [4]. The authors observe that content augmented with meta-data provides a statistically significant gain in learning experience over traditional learning resources.

The learners are provided assessments from a question repository to test their understanding of the concepts. To ensure fair assessments and avoid plagiarism, academicians must ensure the learners receive diverse questions. Enriching the content with meta-data like difficulty levels helps in dynamically changing the questions in assessments based on learner performance [5]. The academicians can also introduce linguistic variations in questions for generating diverse questions for learners to avoid plagiarism, while ensuring that the questions generated also cover diverse concepts. However, generating such assessments manually is a laborious task and a redundant exercise. There is a need for automated difficulty tagging of questions [5] and automated paraphrasing methods for generating such assessments at a large scale. Automated enrichment of assessments with adaptive questions would help learners with the *learning by doing* paradigm, as advocated by the *active learning theory* [6].

In this work, we focus on two major components. We first tackle the problem of automated content curation at scale by identifying and solving various sub-problems. Second, we tackle the problem of enriching existing content with meta-data like granular concepts, difficulty levels and generating diverse content, particularly in the context of assessments conducted to test the understanding of the learners. Our goal is to build assistive tools that can aid academicians in collecting learning content for online platforms.

1.2 Thesis Problem

We summarize the broad vision of the thesis as follows

To build a system that assists the academicians in curation and enrichment of learning content in online e-learning platforms. The system must be modular for efficiency purposes and serve diverse downstream applications.

In the following sections, we identify the sub-problems required to achieve the stated goal.

1.2.1 Automated Content Curation

To onboard content from diverse sources and enable smart search/navigation capabilities, they must be first organized according to a learning taxonomy. The learning taxonomy is usually hierarchical in nature. In this work, the learning taxonomy is of the form *subject - chapter -topic*. The hierarchical content tagging is usually formulated as a multi-label categorization problem. However, it is challenging due to the class imbalance problem at the topic level. Traditional approaches to hierarchical text categorization ignore the hierarchical information in the provided taxonomy, resulting in erroneous or incomplete categorization [7, 8]. There is a need for an approach that can leverage the hierarchical information in the taxonomy while addressing the class imbalance issue for effective content organization. In our work, we propose a module for hierarchical taxonomy tagging of learning content that tackles the mentioned problems by adopting a dense retrieval approach.

The learning taxonomy tagging module aids in automated curation by organizing incoming data to the appropriate level in the taxonomy. However, the taxonomy is limited to the topic level. For better search capabilities, further granular facets like concepts are critical. The concepts are the atomic units users are expected to learn from a given resource.

Annotating the learning resources with fine-grained concepts would aid in keyword-based search and linking related resources. We propose a module for the semi-automated extraction of salient terms that serve as concepts from a given learning resource.

The above sub-problems aid in onboarding new learning content at scale from varied sources. We further elaborate on the contributions in section 1.4.

1.2.2 Content Enrichment

Though learning platforms have made education accessible to learners across demographics, content delivery is passive and does not consider the learning needs of different users [9]. The content delivery can be made more interactive and diverse by linking related knowledge sources. The foundation of this hypothesis is grounded in learning theory. Learning science researchers have proposed several theories based on how different learners interact with the learning content [10, 11]. They observe that *Active learning* [6] helps learners assimilate new information by connecting it to their existing knowledge. Active learning is based on the theory that external knowledge helps users rationalize and create new ideas or update existing ideas. Hence, linking related content and recommending them would help the learner connect related ideas or synthesize new ideas. The original learning resource can be further annotated with latent concepts from external Knowledge Bases (KBs) like Wikipedia to accomplish this. The overlap of related concepts across learning content would then be useful for linking related content.

As discussed in section 1.1, online platforms use assessments to gauge learners' progress. The questions in the learning repository relevant to the topic under consideration can be retrieved for such assessments. However, the e-learning platforms must ensure that the difficulty levels of assessments are consistent for all users. This can be accomplished by adapting the questions according to their difficulty levels based on the learner's progress in the current assessment. However, the questions from third-party sources and the internet are rarely categorized to difficulty levels. Automated question difficulty estimation approaches [5] have been proposed in the past, relying on rule-based systems or statistical features. More recently, deep learning approaches have been proposed [12]. However, the definition of difficulty is subjective and depends on the context of the subject and the grade level of targeted learners. For a reasonable solution, the automated categorization module must be grounded on existing pedagogical rules like the cognitive levels from the Bloom's taxonomy. Further, to generate diverse questions for assessments, we propose the task of paraphrasing Math Word Problems (MWP).

Our primary stakeholders for the proposed system are the academicians who would benefit from the assistance of an automated system. Some downstream applications also help improve the learning experience for the users of the platforms that employ our system. A well-established taxonomy for identifying stakeholders for AI-based interventions for education was established in [13]. We position that our work focuses on the *proactive engagement* of Artificial Intelligence (AI) methods [13] focusing efforts on the curation of content for effective use of resources. We do not directly address the *reactive engagement* component discussed in [13] in our work.

1.3 Challenges

We envision building an intelligent data curation pipeline to assist in effective learning content curation and enrichment. However, unlike existing intelligent tutoring systems, which employ monolithic design [14, 15, 4], the primary challenge here is the need for the decoupling of various functionalities. For instance, one e-learning platform may lack content organization according to difficulty levels but may already have a standardized learning taxonomy. In another scenario, a platform may need fine-granular facets like concepts apart from existing topic-level organization. Hence, there is a need to identify sub-problems to design modules that can exist independently and aid other modules when required.

Secondly, an important challenge is the lack of labeled data or limited labeled data for tasks like identifying fine granular concepts from learning content and content paraphrasing. While supervised feature engineering and deep learning approaches [16] have been applied for tasks like concept extraction, they require a manual time-consuming annotation process. To tackle the data scarcity issues, unsupervised and self-supervised methods [17] have been explored extensively for various retrieval and NLP (Natural language Processing) tasks. The class of self-supervised approaches has been successful, as demonstrated by the utility of Large Language Models (LLMs) [18]. The LLMs are pre-trained on large volumes of web-scale text in a self-supervised fashion where the model learns to predict the next word given a context. Then they are fine-tuned on specific tasks for downstream evaluation. The pre-trained LLMs have demonstrated impressive performance on a variety of retrieval [19] and NLP tasks. However, they also require significant amounts of labeled data when fine-tuning for downstream tasks. This can be tackled by leveraging the dense representations from these models coupled with external knowledge and weak supervision.

The models also have to be computationally efficient when deployed for end users. To this end, we apply knowledge distillation [20], and quantization [21] methods for smaller, efficient versions of the proposed models. However, there is a trade-off between model pruning and the resulting performance. The model efficiency can also be attained by replacing internal mathematical operations in the neural architecture with an efficient version without sacrificing performance compared to model quantization. An example is adopting a linear attention mechanism in transformer models [22] wherever applicable instead of the default attention mechanism, which has quadratic complexity in sequence length.

While Large Language Models have shown impressive performance on a wide range of tasks, they do not seem to understand arithmetic [23, 24] operations. In particular, when applied to the task of paraphrasing, they lose critical information like numbers, units, or other linguistic information necessary to preserve the solution to the original problem as shown in [24]. To tackle this, the language model generation must be controlled to preserve critical information.

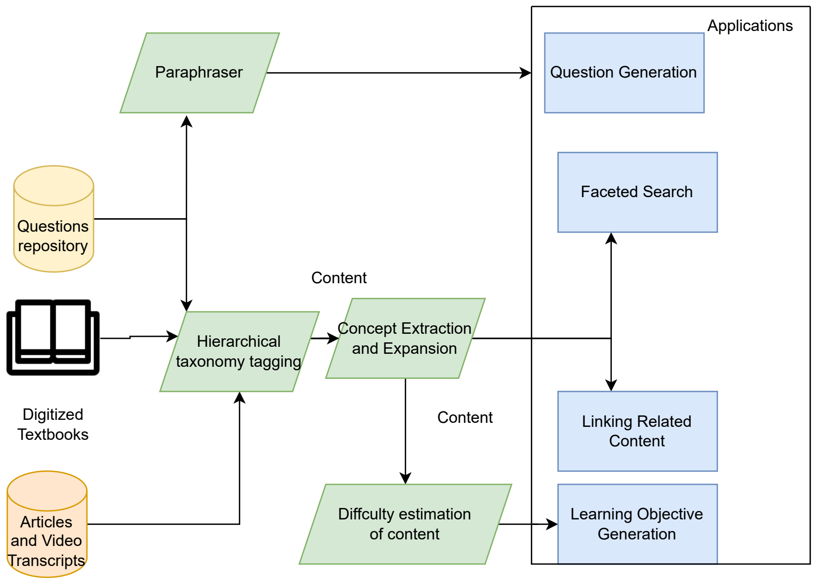
Another challenge is the lack of labeled data for the task of MWP paraphrasing. While well-established benchmarks exist for general domain paraphrasing [25], there is no known dataset for Math Word Problem paraphrasing. To tackle data scarcity, data augmentation approaches [26] are a known alternative. However, text augmentation methods have limited diversity due to the discrete nature of the text. Additionally, these approaches introduce the risk of losing critical information in MWPs.

While we have discussed the technical challenges above, building the proposed system also entails pedagogical challenges. While the systems are powered by systems trained on experts' annotated data in supervised scenarios, self-supervised and unsupervised approaches may produce results not aligned with pedagogical requirements. For instance, difficulty levels assigned to the learning content are subjective and vary from learner to learner. While pedagogical grounding for difficulty levels relies on cognitive levels from Bloom's taxonomy, it may not apply to all subjects. To address such issues, such systems should be deployed with a human in the loop for continuous feedback. The feedback can be incorporated by updating the model in light of new observations.

In this work, we focus on tackling the mentioned challenges to the best of our abilities. We provide a preliminary solution to pedagogical challenges through our applications and by grounding our approaches on the knowledge of domain experts wherever applicable. However, a comprehensive solution to pedagogical challenges requires seamless integration of the proposed system across diverse user groups with a continuous feedback mechanism. This is beyond the scope of the proposed thesis problem. We open-source our approaches to facilitate further work in this direction. We provide a brief overview of potential solutions to the mentioned challenges in section 1.4.

1.4 Thesis Contributions

We first tackle the problem of content curation by developing pipelines that tackle sub-problems like hierarchical taxonomy tagging, fine-grained concept extraction, and ques-



Applications
Paraphraser
Question Generation
Faceted Search
Questions repository
Content
Hierarchical taxonomy tagging
Concept Extraction and Expansion
Digitized Textbooks
Content
Linking Related Content
Articles and Video Transcripts
Diffculty estimation of content
Learning Objective Generation

Figure 1.1: An illustration of the core modules (colored green) and related applications (colored blue) in envisioned system

tion categorization. The core modules and related tasks are illustrated in Figure 1.1. The following topics are the core modules that aid in content curation and enrichment.

1.4.1 Hierarchical Taxonomy based Tagging of Learning Content

Automated Hierarchical Taxonomy Tagging of Content aids in standardizing content in online learning platforms. It also aids in faceted retrieval of content where learners can search through facets like subject, chapter, or topic as discussed in section 1.1.

The class of problems involving categorization of content to labels of hierarchical form are usually cast as flat multi-class classification tasks [7, 8]. The flat classification approaches ignore the hierarchical structure in the label space and encode the labels as numbers. Several approaches [27, 28, 29] consider only the leaf nodes as labels to reduce the label space. In the former method the hierarchy is ignored and in the latter the problem of class imbalance occurs as most of the content is attached to a few leaf nodes. Another challenge is the *open-set identification* problem where new labels may emerge in the label space owing to addition of new topics or removal of old topics. The new hierarchical labels would still be semantically related to the old labels and hence the model must be able to adapt to changes in the label space without re-training. Further, traditional multi-label multi-class classification approaches require changes in the model architecture and re-training to adapt to changes in the label space.

A summary of our contributions for hierarchical taxonomy tagging of content are as follows:

- We reformulate the well-known hierarchical text categorization task to a retrieval task to tackle the discussed problems with traditional approaches. We propose a module to retrieve top-k relevant tags for a given content.
- We propose a novel and efficient contrastive learning approach to learn representations that align the learning content and the appropriate taxonomy in a continuous vector space.
- We study the ability of the proposed approaches to adapt to changes in the label space without re-training or changes in the model architecture. We also perform zero-shot evaluation of the proposed model on diverse data sources to test the ability of the model to organize diverse learning content.
- Our experimental results demonstrate that the gains from the proposed approach are statistically significant when compared to traditional multi-label, multi-class categorization approaches. They are also more efficient than other neural ranking approaches for the taxonomy tagging task.

1.4.2 Fine-grained Concept Extraction using Topic Aware Contextualized Representations

Concepts are the atomic units of any learning content that also drive learning outcomes. Identifying concepts can aid in fine-granular indexing and retrieval, as discussed in section 1.2. It also serves as the backbone for generating learning objectives that guide the learner to the concepts one is expected to learn from a given content. We formulate the problem of concept extraction as a keyphrase extraction task. To tackle the lack of availability of annotated keyphrases, we adopt an unsupervised approach.

Automatic Keyphrase Extraction is a well studied problem [30]. Unlike the supervised methods, the unsupervised methods do not require annotated documents and rely on in-corpus statistical information for extracting keyphrases. In most of the unsupervised keyphrase extraction methods [31, 32, 33, 34] the candidate phrases are represented by a word graph formed based on a co-occurrence window and then ranked. Recent unsupervised methods like EmbedRank [35] leverage representation learning methods that help to capture the semantic relatedness between the phrases and the document.

We propose a novel unsupervised graph-based ranking approach that leverages vector representations from pre-trained model for concept extraction from text. A summary of core contributions for the concept extraction module are as follows,

- We propose a novel vector representation method for phrases. The proposed approach produces topic-aware neural representations from pre-trained language models [36] to cluster together topically and semantically similar phrases.
- We propose two measures *coherence* and *informativeness* for extracting keyphrases that can serve as concepts. The candidate phrases *coherent* if they convey a consistent idea [37] and they are *informative* if they convey the core ideas discussed in the learning content.
- We evaluate the ability of the proposed approach on the task of keyphrase expansion. We demonstrate that the same algorithm can be used for discovering latent phrases/-concepts not present in the given text. We also provide quantitative and qualitative analysis on the ability of the expansion step to link related content.

1.4.3 Interactive Multi-task Label Attention Model for Question Difficulty Estimation

This module aids in simultaneously determining Bloom's taxonomy level and difficulty level of learning content. This can be used to serve adaptive quizzes based on the progress of the learner. Traditional difficulty estimation modules relied on conducting pretests and manually collecting feedback from learners. However, these approaches are prone to exposure bias [5]. Then machine learning approaches were proposed for automated difficulty estimation. They were based on statistical features in text and leveraged classical ML algorithms like Support Vector Machines (SVM) [38]. More recently, deep learning approaches have been employed for the task [12, 39].

We propose a novel approach that leverages the relationship between pedagogically established Bloom's taxonomy levels and the associated difficulty labels. In summary, our primary contributions are:

- We study the relationship between Bloom's taxonomy level and difficulty levels using the Cramer's V test. We observe a significant correlation between the levels, indicating they can benefit from each other. Hence, we adopt a multi-task learning setup to simultaneously predict the Bloom's taxonomy levels and difficulty levels for the questions.
- We propose a label attention mechanism where the model representations are recomputed with respect to the predicted Bloom's taxonomy level. This captures the explicit relationship between the tasks and provides a mutually reinforcing relationship.
- We study the ability of the proposed model to generalize to datasets without the Bloom's taxonomy labels. We achieve this through weakly supervised learning, where our fine-tuned model is leveraged for pseudo-labeling the questions with cognitive levels from Bloom's.

- We observe the proposed approach leads to significant gains over traditional deep learning approaches and multi-task approaches.

1.4.4 Self-supervised Paraphrasing of Math Word Problems

We Focus on MWP paraphrasing in this work to generate diverse problems for learners. The existing approaches that work well on general domain paraphrasing perform poorly on MWPs. Hence, we propose two modules that aid in controlling the output of large LLMs to preserve critical information while paraphrasing. The core contributions are:

- We propose a novel task of Math Word Problem paraphrasing to provide diverse practice questions for learners and to help tackle plagiarism.
- We build a MWP paraphrase quality checker as existing semantic similarity approaches fail to capture loss in critical information. We propose a self-supervised contrastive learning approach that generates positive and negative paraphrases using data augmentation approaches. This module serves as a quality checker which determines is a candidate paraphrase is a valid MWP paraphrase.
- Since MWP paraphrasing does not have annotated data, we propose a self-supervised denoising auto-encoder-based framework for paraphrasing. We propose novel augmentation methods which preserve critical information while facilitating diversity in generated paraphrases.
- We open-source the data to facilitate further work in this area.

1.5 Applications

We built several tools in this work for our partner company named ExtraMarks. We also release open-source versions of these tools to encourage the application of text processing for education and learning science research. The practical contributions of this work are:

- A tool for onboarding new content (https://github.com/ADS-AI/TagRec_Plus_Plus_TKDE.git). The learner can upload any text-based content or batch of content to this tool, which is in turn organized to a hierarchical learning taxonomy using the taxonomy tagging approach. Further fine-grained concepts are extracted from the tagged document and expanded using external knowledge bases like Wikipedia. This is useful for purposes of creating granular indices to aid advanced search and linking related content, as shown in Figure 1.1.

- An end-to-end learning objective generation tool (https://github.com/ADS-AI/CoTagRank_ECIR2022.git. Learning objectives guide the learner to the goals to be accomplished by assimilating knowledge from content. The learning objectives are determined based on the concepts learned and the cognitive complexity required to learn them [40].

- We release the first open-source corpus for K-12 education, which contains digitized versions of NCERT textbooks and learning content from other open-source platforms [41]. We also release a Bidirectional Encoder Representations from Transformers (BERT) model pre-trained on this corpus at huggingface (https://huggingface.co/vasugoel/K-12BERT).

- We develop an unsupervised question duplicate detection tool [42] to de-duplicate large question repositories. This tool leverages concept extraction and learning taxonomy tagging modules. Given an input question, it is tagged to an appropriate taxonomy of form (subject-chapter-topic). This question is then only compared to questions belonging to the same topic. Then the core concepts are extracted using the proposed keyphrase extraction approach for computing a similarity metric between possible pairs of questions. This application benefits from the proposed modular design for the system.

1.6 Thesis Organization

The rest of the dissertation is organized as follows:

Chapter 2 contains a detailed literature review for each module in the proposed pipeline. Chapter 3 explains in detail the proposed approach, experimental setup, and results for hierarchical learning taxonomy tagging of content. The unsupervised extraction and expansion of fine-grained concepts from learning content are covered in Chapter 4, along with related experiments and results. We then provide examples of other modules, like question duplicate detection and question generation, that leverage these components in isolation. We then elaborate on our multi-task modeling approach for Bloom's taxonomy identification and difficulty level prediction of learning content in Chapter 5. We demonstrate on how the modules discussed in Chapters 3,4, and 5 can facilitate the generation of learning objectives. We discuss the workflow of the learning objective generation tool with examples in detail.

In Chapter 6, we discuss the novel task of paraphrasing Math Word Problems to aid in the enrichment and diversification of problem-solving related learning content. The Chapter is divided into two portions discussing the process, experiment, and results of the paraphrasing quality checker and the paraphraser. We also demonstrate the intelligent paraphrasing tool with examples at the end of the chapter.

Finally, in Chapter 7, we conclude by providing a summary of the thesis contributions and their limitations. We also discuss future work to facilitate further research in this area.

CHAPTER 2
BACKGROUND AND LITERATURE REVIEW

Online learning platforms have gained tremendous popularity in recent years as they enable remote education [1]. However, the abundance of learning resources renders it difficult for the learners to access the relevant content. To enable ease of access to the content, proper organization of content is pivotal to enable smart search and navigation capabilities. For content curation and design, a large body of work uses traditional systemic approaches [2]. In recent times, AI-assisted content design [1, 13] has gained enormous interest and shown significant gains in efficiency. In the survey, [13] on AI-assisted learning, the authors identify two stages for technology intervention in learning platforms. The *proactive engagement* where AI technologies are leveraged for content curation and curriculum design. The intervention at a later stage is termed *reactive engagement* where the related technologies are used to predict student outcomes, assess performance, and deal more with user personalization [43, 44, 45, 46, 47]. In this dissertation, we focus on the *proactive engagement* by building tools for automated content curation and enrichment in online learning platforms. While there have been numerous works on *reactive engagement*, such as predicting student outcomes [47], very few works have focused on content curation and enrichment. Most works on content design have focused on augmenting the learning content with meta-data such as difficulty levels [16, 13, 5, 48]. The approaches used for this task range from classical rule-based techniques to more recently deep learning based approaches [12]. Few works have focused their efforts on the content organization in learning platforms. Most of these works revolve around extracting fine-grained concepts for indexing, and linking related content [49, 50, 51]. However, to curate and onboard content at scale, applications for automated management of learning taxonomy and tagging are critical. In this work, we identify such applications and envision building an end-to-end system that can aid academicians in curating content at scale from various sources. While the applications are not exhaustive, we identify and build fundamental modules that are de-coupled and can be used for numerous downstream applications.

In the following sections, we discuss prior work for the different components identified in Chapter 1.

2.1 Hierarchical Label Categorization Methods

In this section, we discuss the core approaches adopted for categorizing different types of content to labels of hierarchical format.

2.1.1 Text categorization to Hierarchical Labels

The online systems use a standardized taxonomy to organize their content [7, 8]. The taxonomy is of hierarchical nature, and usually, the approaches used to categorize content in such taxonomy can be categorized into multi-class classification or hierarchical multi-step approaches [52, 53]. In multi-class single-step methods, the leaf nodes are considered labels while ignoring the hierarchy. This leads to class imbalance issue where a large number of samples cover only a few leaf nodes considered as labels. In the hierarchical multi-step approach, the root category is predicted using a classifier, and the process is repeated to predict the nodes at the subsequent level. However, the main issues of the approach are that the number of classifiers increases with depth, and the error from one level propagates to the next level. This also increases the computation needed at inference time. Along similar lines, Banerjee et al. [54] proposed to build a classifier for each level. However, unlike the previous works, the parameters of the classifier for parent levels are transferred to the classifiers at child levels. Another approach [55] proposed to use a chain of neural networks to categorize content to hierarchical labels. A classifier is designated for each level in the hierarchy. However, the major limitation here is that the number of networks in the chain increases with depth, and it also requires that the paths in the label hierarchy should be of the same length, limiting the applications to cases of minor changes in the label space.

To circumvent these issues, each hierarchical taxonomy could be considered as a label disregarding the hierarchy, and a regular single-step classifier could be trained to classify the content to one of the labels. Several single-step classifiers have been proposed for classification tasks involving hierarchical labels. In [53], the word level features like n-grams were used with SVM as a classifier to predict level 1 categories, whereas in [8] the authors have leveraged n-gram features and distributed representations from Word2Vec followed by a linear classifier for multi-class classification. Several deep learning methods like CNN [56] and LSTM [52] have been proposed for the task of question classification. Since the pre-trained language models, like BERT [57], improve the performance, the authors in [7] propose a model BERT-QC, which fine-tunes BERT to categorize questions from the science domain to labels of hierarchical format. The problems involving hierarchical labels have also been formulated as a translation problem in [58] where the product titles are provided as input and use a seq2seq architecture to translate them to product categories having hierarchical structure. The hierarchical neural attention model [59] leverages attention to obtain useful input sentence representation and uses an encoder-decoder architecture to predict each node in the hierarchical learning taxonomy. However, this approach may not scale as the depth of the hierarchy increases.

Several works have also tried to capture the hierarchical structure of the labels for the purpose of text categorization to such labels. For instance, Zhou et al. [60] proposed to design an encoder that incorporates prior knowledge of label hierarchy to compute label representations. However, they flatten the hierarchy and treat every label as a leaf node which would require re-training when there are changes in the label space. Lu et al. [61] introduced different types of label graphs (co-occurrence based and semantic similarity based) to improve text categorization. However, they also cast the task of categorizing text to hierarchical labels as a multiple binary classification task [62]. These approaches do not consider the relationship between the terms in the text inputs and the hierarchical labels.

2.1.2 Sentence representation methods

The NLP tasks like classification and retrieval have been advanced by distributed representations that capture the semantic relationships [63]. Methods like GloVe [64] compute static word embeddings that do not consider the context of the occurrence of the word. An unsupervised method named Sent2Vec [65] was proposed to create useful sentence representations.

The Bidirectional Encoder Representation from Transformers (BERT) [57] does not compute any independent sentence representations. To tackle this, Sentence-BERT [36] model was proposed to generate useful sentence embeddings by fine-tuning BERT. Another transformer-based sentence encoding model is the Universal Sentence Encoder (USE) [66] that has been specifically trained on Semantic Textual Similarity (STS) tasks and generates useful sentence representations.

In this paper, sentence representation methods are used to represent the labels by treating each of them as a sequence. For example, the label **Science - Physics - alternating current** is treated as a sequence. We employ sentence representation methods to model the compositionality of terms present in the hierarchical learning taxonomy. Several works [67] [68] have demonstrated that sentence representation models are able to capture the nature of how terms compose together to form meaning in a sequence. We posit that the same principle can be used to capture the complete semantic meaning of hierarchical taxonomy in the vector space. In our experiments, we observe that sentence transformer based representation methods perform better than averaging word embeddings.

2.2 Automated Keyphrase Extraction

The huge growth of textual data on the web has lead to the development of novel Automatic Keyphrase Extraction (AKE) methods to extract relevant information for downstream tasks like text classification, information retrieval, and other tasks like document indexing. A detailed survey of AKE methods is outlined in the study [30]. The Automatic Keyphrase Extraction methods can be categorized as supervised and unsupervised methods. The unsupervised methods are usually preferred for extracting keyphrases as they require no labeled data and are less resource-intensive when compared to the supervised methods. Here, primarily the unsupervised class of keyphrase extraction methods are discussed. The unsupervised methods usually operate in three steps:

- The candidate keywords are extracted from the document. Usually, nouns and adjectives are chosen as candidate words.
- Then, the candidate list is ranked using a ranking algorithm. The first step in ranking is to represent the words in a way such that it captures the syntactic and semantic relationship with other candidate words. The representation mechanism depends on the underlying approach. The relationship between different words can be defined using the local *statistical* information like how frequently they co-occur in the given document [34, 69]. Other approaches include leveraging vector representation methods where the words are projected to a continuous vector space [35]. Then the semantic relationship between the words can be captured as the cosine similarity between their vector representations in the continuous vector space.
- After the keywords are ranked, the top keywords are grouped to form keyphrases.

Though the above steps are generally adopted in most unsupervised keyphrase extraction methods, some methods like EmbedRank [35] directly extract candidate keyphrases and rank them instead of ranking keywords. The unsupervised keyphrase extraction methods are discussed in detail in the following paragraphs.

Graph based ranking methods are very popular among the unsupervised keyphrase extraction algorithms. The graph based keyphrase extraction methods were introduced in the seminal work TextRank [34]. It constructs a weighted graph for the given text where the nodes are represented by words, and an edge connects word types only if they co-occur within a window of specified size. The fundamental idea here is that each node contributes a *"vote"* to the node it is linked to. The votes for a node from multiple nodes linked to it are aggregated, and this aggregated score can be seen as the *importance* of a node. Additionally, the significance of a *"vote"* cast by a node is determined by the *importance* of the node casting the vote. Hence, by iteratively aggregating the *"votes"* for each node using Google's *personalized PageRank* algorithm, a score is obtained for each node. The algorithm is run till the scores obtained for the nodes do not change between subsequent iterations. Once this convergence criterion is reached, the scores obtained for each node are used to sort the vertices in reverse order. Then from the sorted list, top-k vertices are selected, where k is a parameter specified by the user. The top-k vertices correspond to top-k keywords from the given text, which can be used for downstream tasks like indexing. Some of the keywords are also combined to form keyphrases. The authors also demonstrated that the proposed algorithm could also be used for sentence selection, where the nodes of the graph are formed using sentences instead of words. Then the sentences are ranked using the mentioned algorithm, and top-k sentences are selected. Though the results given by TextRank were promising, it suffered from one issue. The edges in the graph formed by TextRank were uniformly weighted. But this is not desirable, as it may result in unimportant words being assigned a higher score just by virtue of their frequent occurrence in the given text. Additionally, some edges that connect two frequently co-occurring or rare words could be assigned a higher weight to ensure it appears in top-k results.

The SingleRank [69] algorithm was an extension to the TextRank algorithm, where the edges were assigned a weight equal to the co-occurrence count of different words in the vocabulary. The authors show that it leads to a better ranking of words when compared to the TextRank algorithm, where all the edges between nodes were assigned uniform weights. Additionally, in the SingleRank algorithm, after the nodes in the graph are assigned scores using the PageRank ranking mechanism, the low scoring nodes are not filtered out like in the TextRank algorithm. Rather, the keyphrases are created by grouping matching keywords, and the score for each keyphrase is computed as the sum of the scores of the constituent keywords. Then the keyphrases are ranked using their scores, and top-k keyphrases are obtained as output.

The unsupervised keyphrase extraction methods discussed so far leverage only local *statistical* information and do not consider the semantic information of the words in the document. This limitation leads to less informative keyphrases being extracted from the document. The WordAttractionRank [70] algorithm was proposed to address this issue. The authors propose two measures, namely the *informativeness* and *phraseness* measures, for a phrase to be considered as a keyphrase. The authors posit that a candidate phrase is considered to be an *informative* keyphrase if it conveys the main ideas of a document. This implies that the constituent words of a phrase must be semantically related and convey the main ideas of the document. Hence, the authors formulate the *informativeness* measure as the product of frequencies of individual words divided by the euclidean distance between the vector representations of the words. Since words that frequently co-occur can be grouped to form a meaningful phrase, the authors formulate the *phraseness* measure as the co-occurrence information of the words. In the WordAttractionRank algorithm, a weighted undirected graph is formed where the *informativeness* and *phraseness* measures are used to weigh the edges. The ranking procedure is similar to the SingleRank [69] algorithm.

The unsupervised keyphrase extraction methods discussed so far work well on short and medium length documents like scientific abstracts. However, on long documents like full length scientific papers, they fail to extract informative keyphrases. This is because the number of candidate keywords in the long documents is huge, and it is difficult to identify meaningful co-occurrence information between words. To circumvent this issue, several approaches like PositionRank [71], and MultipartiteRank [72] which leverage positional information of various candidate words to extract informative keyphrases have been proposed. As keyphrases usually appear at the beginning of the document, the PositionRank [71] algorithm and the MultiPartiteRank [72] algorithm assigns weights to nodes (words and phrases respectively), favoring the terms appearing at initial positions in the text. The graph formation in the PositionRank algorithm is similar to the previous algorithms discussed. The nodes are represented using the words, and the edges are weighted using the co-occurence count of the words in the given document. Additionally, the PositionRank algorithm specifically weighs each word by the inverse of its position in the document. Hence the words that appear early in the document are assigned higher weights than the words occurring later in the document. The MultiPartiteRank algorithm also uses positional information like the PositionRank algorithm with differences in the way the graph is formed. In the MultipartiteRank algorithm, a directed graph is formed in which the nodes are represented using candidate keyphrases instead of words. The second difference, when compared to traditional graph based algorithms, is that a multi-partite graph is formed where two candidate keyphrases (nodes) are connected only if they belong to different topics. The paper does not make any assumption regarding the topic identification method used, and hence any of the existing state-of-the-art methods like Latent Dirichlet Allocation (LDA) [73] can be employed. Then the weights of the first candidate keyphrases in each topic are adjusted by multiplying the inverse of their position number in the document. In both of the works (PositionRank and MultiPartiteRank), the authors observe that the quality of the extracted keyphrases is better than other approaches that do not leverage positional information.

Several existing approaches have leveraged topical information for ranking phrases. The TopicRank algorithm [31] proposed the representation of nodes in the graph as topics instead of words. Another algorithm that leverages the topical information is the Topical Page Rank (TPR) [32] algorithm. The TPR method runs TextRank for each topic, where the topics are obtained using LDA [73]. The final scores of the candidate keyphrases are obtained by the sum of scores of keyphrases across topics.

One of the shortcomings of these approaches (except MultiPartiteRank) is that they rank the phrases by aggregating the scores of the constituent words. This can lead to uninformative candidate phrases being ranked higher just because one of the constituent words has a higher score.

In contrast to the graph based methods, the EmbedRank [35] algorithm is an embedding based unsupervised keyphrase extraction method that represents both documents and candidate phrases as vectors in a continuous vector space using document embedding methods like Sent2Vec [74]. This work pioneered the use of sentence based vector representation methods for the task of keyphrase extraction. The vector representations help to rank candidates by computing cosine similarity between the phrase vectors and the document vector. The authors devise two measures, namely the *"informativeness"* and the *"diversity"* to extract high-quality keyphrases. The *"informativeness"* measure is defined as the semantic relatedness between the phrase and the document vector representations. The *"informativeness"* measure helps filter out unimportant phrases, and only phrases that have high "informativeness" scores are retained. The *"diversity"* measure is defined as the semantic relatedness between the phrase vector representations. This measure helps filter out redundant keyphrases. However, the EmbedRank algorithm does not leverage the PageRank algorithm like existing unsupervised methods and rather relies only on the semantic relatedness between the vector representations of the candidate phrases extracted from the document. A graph based ranking procedure augmented with the semantic relatedness between phrases as weights could help produce more diverse phrases due to the inherent randomness present in the PageRank algorithm. The Key2Vec algorithm [75] proposed by Mahata et al. combines the benefits of the distributed phrase representations and the PageRank based ranking method. The algorithm represents the candidate phrases using the vector representation methods like Fasttext [76] and then forms a graph of candidate phrases where the edges are weighted by the semantic similarity computed between the phrase representations of the corresponding words. Then the nodes are ranked as usual using the personalized PageRank algorithm. The authors applied the algorithm for keyphrase extraction from scientific documents and observe an increase in precision and recall when compared to the existing state-of-the-art methods.

Hence, the task of keyphrase extraction has a huge impact on applications like document indexing and search. They also provide a concise summary of the document and help the readers. The unsupervised keyphrase extraction methods require less compute when compared to the supervised counterparts and require no labelled data. The unsupervised algorithms mainly leverage the local statistical properties of keywords occurring in a document and rank the words using a graph based algorithm like PageRank. Then the graph based ranking methods were improved to rank the phrases directly instead of words in order to avoid unimportant phrases from being ranked higher. Further improvements involved leveraging pre-trained neural network models like Sent2Vec [74] to represent the candidate phrases in a continuous vector phrase. These representations helped in capturing semantic relatedness between phrases rather than just the local statistical properties, resulting in the extraction of high quality keyphrases. But these methods do not use positional information, leading to poor performance on long documents. Further improvements to the unsupervised methods could involve combining positional information and the vector representation methods to generalize the class of methods to long documents instead of using them in isolation. Additionally, contextualized sentence representation methods like SentenceBERT [36] or Universal Sentence Encoder [66] could be used in methods like EmbedRank instead of static vector representation methods like Sent2Vec. The contextualized vector representation methods provide different vectors for the same word depending upon the context of its occurrence and hence helps capture the meaning of a word or a phrase in context of its nearby words in the sentence. This would help in extracting high quality keyphrases.

2.3 Difficulty and Bloom's taxonomy Estimation

Question difficulty prediction is an intriguing NLP problem, but it has not been explored to the extent it deserves. A few recent works in difficulty prediction focuses on evaluating the readability of the content[77, 78]. Authors in [78] proposed to combine classical features like Flesch readability score with non-classical features derived automatically. In [77], the authors observed that combining a large generic corpus with a small population specific corpus improves the performance of difficulty prediction.

Another application where the task of question difficulty prediction has been discussed is the automated question generation[79]. In the work [79], the authors propose to leverage the difficulty level of a question for evaluating the distractor (wrong answer options) prediction task when generating MCQ questions. The authors estimate the difficulty measure as the semantic similarity between the question and the answer options. The intuition behind this approach is that the MCQs should have good distractors, which are similar to the correct answers, making it difficult for the student to guess the correct answer. Similarly, in the work [80], the authors propose a similarity based theory for controlling the difficulty of the questions. Though these works propose a good set of features for difficulty prediction, they do not explore training on related tasks that may improve the performance. We explore leveraging multi-task learning based approaches by relating Bloom's taxonomy with difficulty levels to improve performance on the difficulty prediction task.

Prior work has also focused on estimating the difficulty of questions in community question-answering (QA) platforms like StackOverflow [81, 82]. Liu *et al.*[81] propose a competition based approach that jointly models the difficulty of the questions and the expertise of the users in community based QA services. Wang *et al.*[82] proposed a novel method that considers the textual description of the questions and question-user expertise comparisons as input. It further handles the issue of cold start estimation, where K-nearest neighbor methods are used to estimate the difficulty of the unresolved questions in such communities.

Other studies like [39, 83] explore automated grading by estimating the difficulty of academic questions from the Science domain. Pado *et al.*[39] perform a detailed analysis of the data and show that there is a strong correlation between the Bloom's taxonomy levels and the difficulty levels. In the work [83], the authors propose to infer the difficulty of the questions by first mapping them to the concepts. The authors propose a multi-task convolutional neural network for jointly predicting the topics and concepts for the questions. In the work [12], the authors propose fine-tuning the ELMo[84] model on 18,000 MCQ type questions from the medical domain for predicting the question difficulty.

Though the above works are relevant, they are not directly comparable with the task proposed in this paper. Unlike the community QA platforms, which contain detailed textual description regarding the question asked, our dataset contains only short questions. The semantic information contained in the question text is low when compared to the questions in the community QA platforms. Our task is also more challenging than the work [79], which deals with only MCQ questions. Since MCQ questions also contain distractors, they serve as a good indicator of difficulty. However, many questions in our dataset are subjective or fill in the blank type questions with no options. Hence, it poses a challenge as the difficulty has to be estimated based on the question and answer text alone.

2.4 Unsupervised and Self-supervised Paraphrasing Approaches

In our work, for enrichment of existing content like questions, we focus on self-supervised paraphrasing methods. In this section, we first briefly discuss prior work in text data augmentation methods and paraphrase identification methods. Then we discuss the self-supervised and unsupervised paraphrasing approaches, since in our work we adopt a self-supervised approach for paraphrasing MWPs.

2.4.1 Text Data Augmentation

Though data augmentation is very common in the Computer Vision domain, little work has been done in the context of text-based tasks. One of the few notable initial works [85] replaced words and phrases with synonyms to obtain more samples for text classification. However, this augmentation provides a minimal improvement in classification. In the work [86], the authors propose noising methods for augmentations where words are replaced with alternate words based on unigram distribution. However, the noising probability is a parameter that controls the quality of the augmentations. A much easier text augmentation method, EDA, was proposed in the work [87]. The authors propose several operators such as random word deletion and synonym replacement to generate new sentences. The number of words to be deleted or replaced is controlled by a hyperparameter α. The above works are based on heuristics and depend on a hyperparameter for high-quality augmentations.

More recently, self-supervised text augmentation methods have provided a superior performance on multiple tasks. In the work [88] the authors propose two text augmentation operators, namely backtranslation and TF-IDF based word replacement, where words with low TF-IDF scores are replaced. Another self-supervised data augmentation method proposed recently was SSMBA [26]. The authors of this work propose a manifold based data augmentation method where the input sentences are projected out of the manifold by corrupting them with token masking followed by a reconstruction function to project them back to the manifold. A separate self-supervised augmentation method named InvDA (Inverse Data Augmentation) was proposed in Rotom [89] which was similar to SSMBA in that it tried to reconstruct the original sentence from the corrupted version. Applying the works mentioned directly to algebraic questions is not as beneficial, as they are sensitive to small changes. The text augmentation operators discussed would not be able to generate negative samples for the task of paraphrase quality detection that quantify the loss in information like numerical entities or units being removed. Even when some of the operators like random deletion from EDA are applied to generate positive samples, they may result in loss of crucial information, rendering the question unsolvable. Several rule-based text augmentation methods have been proposed, like [90] which uses a Natural Language Inference (NLI) guided augmentation and [91] leverages linguistic knowledge for the question-answering task. They show better performance compared to generic text augmentations.

2.4.2 Sentence Representations and Paraphrase Identification

We briefly discuss the work in the area of sentence representation methods and paraphrase identification. Sentence representation methods have seen a surge in recent times and serve multiple applications like sentence pair scoring tasks or retrieval. One of the earliest works in this area was Sent2Vec [65] which proposed to compose n-gram embeddings to learn sentence representations. For pairwise sentence scoring tasks like paraphrase identification, cross-encoders like BERT [57] have been leveraged where the sentences are separated using a separator token and encoded jointly using self-attention. However, cross-encoders are computationally expensive as they do not yield independent embeddings for the sentences and have quadratic complexity due to the combined pairs of sentences. To circumvent this, SBERT [36] uses a bi-encoder architecture where BERT is applied separately on each sentence followed by a pooling layer over the output to yield fixed-length sentence embeddings for each sentence. Then the sentences can be compared during cosine similarity between their embeddings. They demonstrated impressive performance on paraphrase identification tasks using the MRPC corpus [92]. The polyencoders [93] were proposed to combine the advantages of both bi-encoders and cross encoders. However, polyencoders have limited applicability as their scoring function is not symmetric rendering it impractical for paraphrase quality detection tasks. Recently Augmented SBERT [94] was proposed for pairwise scoring tasks. Though the Augmented SBERT tackles the issues of polyencoders, it requires gold standard data as a seed set limiting the applicability to our task. Moreover, existing tasks are concerned with high performance on the positive class (identifying as paraphrases). Our work is concerned with quantifying information loss and identifying negative and positive paraphrases in a balanced manner.

2.4.3 Self-supervised and Unsupervised Paraphrasing Approaches

Since our task of paraphrasing MWPs while preserving the solution is a new task, we discuss closely related self-supervised approaches in this section.

Paraphrasing general domain corpus is a well studied problem and can be categorized into supervised and unsupervised approaches. The supervised approaches have evolved over the years. Earlier approaches [95, 96] leveraged syntactic features to aid in generation of paraphrases. Several approaches framed the task of paraphrasing as a machine translation task and employed statistical machine translation approaches [97, 92]. More recently, the task of paraphrasing has been modeled as a sequence generation task using an encoder-decoder setup comprising deep neural networks. For instance, recurrent architectures like stacked LSTM [98] have been employed for generating paraphrases from input in a supervised setting. More recently, with the success of self-attention for various tasks, the transformers [99] are being used for paraphrase generation. These approaches require labeled paraphrases for training, which requires time-consuming annotation process.

To tackle the problem of paraphrasing without labeled samples, unsupervised approaches have been proposed. The Variational Auto Encoders (VAE) aid in text generation by directly sampling from the latent space [100, 101]. However, the sampling is stochastic and controlled generation becomes difficult during the decoding phase sometimes resulting in incorrect sentences. To resolve this several approaches have proposed data augmentation operators for generating self-supervised corpus to aid in constrained paraphrase generation. Some of the earliest data augmentation approaches proposed are Unsupervised Data Augmentation (UDA) and Self-supervised Manifold Based Augmenttaion (SSMBA). The UDA leverages several operators like backtranslation and word replacement to generate new samples for text classification or generation. The authors of

SSMBA [102] propose a manifold based augmentation methods where words in the input text are randomly masked to project the input out of the manifold and are re-projected to a different point in the manifold by sampling from latent space. Alternate approaches where heuristic search based methods like simulated annealing are leveraged [103] to search over edit operators have been proposed. Since the generated samples are not diverse enough in the above approaches, Rotom [89] proposed diverse operators like synonym replacement, word insertion to perturb the text and generate diverse sentences to train a generative model. However, these approaches still fail to generate diverse paraphrases and also fail to generate semantically equivalent paraphrases for Math Word Problems while preserving the solution.

CHAPTER 3
HIERARCHICAL TAXONOMY BASED TAGGING OF LEARNING CONTENT

3.1 Introduction

In this chapter, we tackle the problem of organizing learning content to a standardized taxonomy [1]. Existing approaches that formulate the task as a multi-label text classification task suffer either from error propagation from coarse levels of the taxonomy or class imbalance at the leaf level [52, 53, 7, 8]. Some of these approaches also fail to capture the structure in the hierarchical labels (taxonomy). We formulate the problem as a dense retrieval task and adopt a contrastive learning approach. We project the content and taxonomy to a continuous vector space by using transformer based encoder models. We propose to learn an alignment between content and taxonomy vector sub-spaces using a hinge rank loss (triplet loss). Since the phrases in the hierarchical labels are abstractions of their token descriptions they are semantically related to the terms in the input content. Hence to induce the structure and meaning of the hierarchical labels in the input content, we propose an efficient cross-attention mechanism that captures the information in the hierarchical labels. In summary, the core contributions of our work are:

- We cast the classical hierarchical multi-label text classification task as a dense retrieval task adopting a contrastive learning approach.
- We propose an efficient cross-attention mechanism to fuse the information in the hierarchical label representations and the content representations. We also provide theoretical bounds where the difference in attention scores are capped by the euclidean distance between the label representations.

[1]This chapter is partly a reproduction of a paper published at European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD) [**tagrec**] and the subsequent extension under review at IEEE Transactions on Knowledge and Data Engineering (TKDE) [104]

Table 3.1: Some samples from the QC-Science dataset

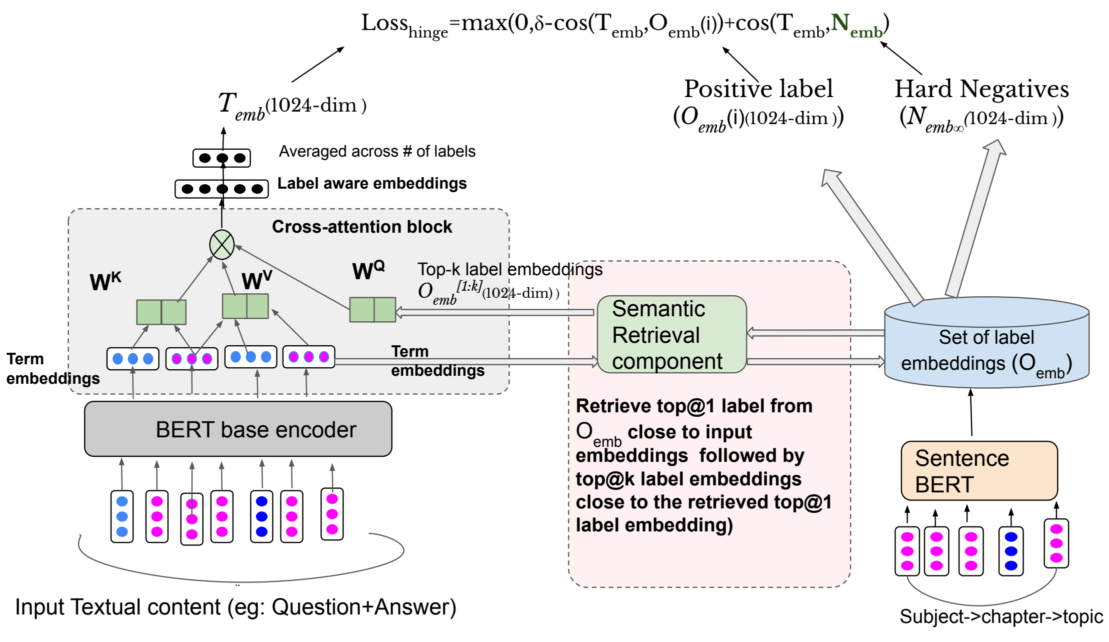
Question	Answer	Taxonomy
The value of electron gain enthalpy of chlorine is more than that of fluorine. Give reasons	Fluorine atom is small so electron charge density on F atom is very high	Science$\rightarrow$chemistry$\rightarrow$classification of elements and periodicity in properties
What are artificial sweetening agents?	The chemical substances which are sweet in taste but do not add any calorie	Science$\rightarrow$chemistry$\rightarrow$chemistry in everyday life

- We propose a novel in-batch adaptive hard-negative sampling approach during training to aid in high-recall retrieval.
- We demonstrate the ability of our model to adapt to new learning content and also to changes in the label space.
- We release the code and data at https://github.com/ADS-AI/TagRec_Plus_Plus_TKDE

3.2 Methodology: An Efficient Taxonomy Aware Encoder for Automated Taxonomy Tagging

In this section, we describe our method *TagRec++* for retrieving labels of hierarchical format for learning content like question-answer pairs. The proposed approach adopts contrastive learning to align the vector sub-spaces of content and hierarchical label representations. An efficient cross-attention mechanism between the content and label representations makes the input content representations taxonomy-aware. We also propose a dynamic hard-negative sampling approach to improve recall of retrieved labels. Figure 3.1 provides an intuition for the proposed approach.

The input to the method is a corpus of documents, $C = \{D_1, D_2 ... D_n\}$. The documents are tagged with hierarchical labels $O = \{(S_1, Ch_1, T_1), (S_2, Ch_2, T_2) ... | S_i > Ch_i > T_i\}$ where S_i (root node), Ch_i and T_i (leaf node) denote subject, chapter, and topic, respectively. The goal here is to learn an input representation that is close in the continuous



$\text{Loss}_{\text{hinge}}=\max(0,\delta-\cos(T_{\text{emb}},O_{\text{emb}}(i))+\cos(T_{\text{emb}},\mathbf{N}_{\text{emb}})$
T_{emb}(1024-dim)
Positive label (O_{emb}(i)(1024-dim))
Hard Negatives (N_{emb_∞}(1024-dim))
Averaged across # of labels
Label aware embeddings
Cross-attention block
W^K
W^V
W^Q
Top-k label embeddings $O_{emb}^{[1:k]}$(1024-dim))
Term embeddings
Term embeddings
Semantic Retrieval component
Set of label embeddings (O_{emb})
BERT base encoder
Retrieve top@1 label from O_{emb} close to input embeddings followed by top@k label embeddings close to the retrieved top@1 label embedding)
Sentence BERT
Input Textual content (eg: Question+Answer)
Subject->chapter->topic

Figure 3.1: Proposed approach: TagRec++

vector space to the correct label representation. We consider the label (S_i, Ch_i, T_i) as a sequence $(S_i + Ch_i + T_i)$ and obtain a sentence representation for it using pre-trained models. Since the sentence representation encoder is frozen during training, we can precompute and index the embeddings for the hierarchical labels. This also ensures faster inference. We leverage BERT [57] for obtaining contextualized embeddings followed by a linear layer. The linear layer maps the 768-D representation from BERT to the 1024-D or 512-D vector representation. The novelty of the method lies in the implementation of the attention layer and in-batch hard negative sampling, which are discussed in detail later in the Section. The steps of the proposed approach can be observed from Algorithm 1. During the *training phase*, the input text is cast to a continuous vector space using a BERT [57] base model. The labels are projected to a continuous vector space using a Sentence-BERT model to capture the composition of terms in the label. To capture the interaction of terms in the content with hierarchical labels, we fuse the information through an attention mechanism between the embeddings of related hierarchical labels and the term embeddings of the content. Then we use a triplet loss objective with in-batch hard-negative sampling approach to pull apart irrelevant labels for a given input content.

Algorithm 1 Tag Recommender

```
Input: Training set $T \leftarrow$ docs $\{D_1, ..D_n\}$, labels $O$ of form (Subject-Chapter-Topic)
Output: Set of tags for test set , $RO$
    Training (batch mode)
 1: Get input text embeddings , $T_{emb} \leftarrow BERT(D)$
 2: Obtain label embeddings, $O_{emb} \leftarrow SENT\_BERT(O)$
 3: $Index(labels) \leftarrow O_{emb}$
 4: Get label embedding closest to input,
    $O^i_{emb} \leftarrow top_1(cos(T_{emb}, Index(labels)))$
 5: Get top labels close to selected label,
    $L_r \leftarrow ||O^i_{emb} - index(labels)||_2$
 6: Project labels to queries, $Q \leftarrow W^Q * L_r$
 7: Project input embeddings to key and values,
    $K \leftarrow W^K * T_{emb}$ and $V \leftarrow W^V * T_{emb}$
 8: Compute modified input embeddings,
    $T_{new} \leftarrow \frac{Softmax(Q * K^T)}{\sqrt{d_k}} * V$
 9: $hard\_neg \leftarrow top\_k(cos(T_{new}, Index(labels)))$ excluding $label$ (adaptive hard-negatives)
10: $L(text, l) \leftarrow \sum_{j \neq l} max(0, \delta - cos(T_{new}, v(l)) + cos(T_{new}, v(j)))$ where $v(j) \in hard\_neg$
11: Fine-tune BERT to minimize $loss$ and align $T_{new}$ and corresponding label representations from
    $O_{emb}$
    Testing Phase
12: Compute embeddings for test set $S$ using fine-tuned BERT $S_{emb} \leftarrow BERT(S)$
13: $RO \leftarrow sorted(Sim(S_{emb}, O_{emb}))$, gives ranked set of labels
14: return Top-k labels from $RO$
```

3.2.1 Efficient Interactive cross-attention module for taxonomy aware input embeddings

The primary goal of the proposed method is to align the vector representations of the input and the relevant hierarchical taxonomy labels. However, their vector representation sub-spaces may not be close to each other. This renders the alignment of the sub-spaces of the input representations and the corresponding label representations a hard problem. However, the tokens in the labels are related to terms in the content. The alignment approach would benefit from capturing the semantic relatedness between various hierarchical labels and a given content. The final vector representations are computed by fusing the information from hierarchical labels and the input. This would result in representations that are aware of the taxonomy and hence could help in performance improvement of the alignment task.

To achieve this, the proposed method first retrieves the label embedding closest to the input representation to capture the interaction between the labels and content which are related to each other (step 4 in Algorithm 1). This will reduce the noise induced by unrelated labels when fusing information from the input and the labels to compute *taxonomy aware* input representations.

$$T_{emb} = f_\theta(D_i)$$

$$O_{emb} \leftarrow g_\theta(O)$$

$$O^i_{emb} \leftarrow top_1(cos(T_{emb}, O_{emb}))$$

where f_θ is BERT and g_θ refers to sentence representation methods like Sentence BERT or Universal Sentence Encoder (USE). As this process occurs in the training loop, the closest label selected depends on the model parameters. The selection process is thereby dynamic and improves with updates to model parameters. When the model gets better at aligning the input and label representations, it will sample better top-1 hierarchical label closer to the input. Our task is modeled as a dense retrieval approach thereby, our goal is to improve the quality of the first label retrieved.

The step-5 in Algorithm 1 retrieves the top-k labels closest to the selected label. We do not compute attention with respect to all labels in the label space to obviate interference from unrelated labels and reduce the computational complexity of attention. We sample a small set of labels that are related to the label closest to the content representations at a given time in the training loop. This step helps to cluster similar labels which are closer to the input and to each other in the vector space. Also, it reduces the complexity involved in computing attention between the input and label representations. The labels sampled approximate the attention distribution well as shown below, adapting the property of attention from the work [105].

The difference between the attention of inputs (Keys) and labels (Queries) can be bounded by the euclidean distance between the labels.

Statement: Given two label representations O^i_{emb}, O^j_{emb}, the difference between attention can be bounded as by the euclidean distance between the label representations.

To arrive at this result, we first start with the principle of Lipschitz continuity for Softmax. Given two queries Q_i and Q_j,

$$||Softmax(Q_iK^T) - Softmax(Q_jK^T)||_2)$$

$$\leq \epsilon||Q_iK^T - Q_jK^T||_2$$

Let ϕ denote the Softmax operation for the rest of the section.

The Softmax operation has a Lipschitz constant less than 1[106] which implies ϵ equals 1. Hence the attention approximation is bounded by the euclidean distance between the queries

$$||\phi(Q_iK^T) - \phi(Q_jK^T)||_2 \leq ||Q_i - Q_j||_2||K||_2 \tag{3.1}$$

Since $Q_i \leftarrow W^Q * O^i_{emb}$ we can write the above equation as:

$$||\phi(Q_iK^T) - \phi(Q_jK^T)||_2 \leq ||O^i_{emb}W^Q - O^j_{emb}W^Q||_2||K||_2 \tag{3.2}$$

Since the norm of the weight matrix W^Q is the largest eigenvalue of $((W^Q)^TW^Q)^{1/2}$ we modify the above equation as

$$||\phi(Q_iK^T) - \phi(Q_jK^T)||_2) \leq ||O^i_{emb} - O^j_{emb}||_2||W^Q||_2||K||_2 \tag{3.3}$$

By equation 3, the difference between the attention can be hence bound as the euclidean distance between the label representations.

Following this bound, step-5 of Algorithm 1 samples top-k hierarchical labels (L_r) based on their proximity to the top-1 label in the representation space.

The input representations are projected to (K)ey and (V)alue matrices.

$$K \leftarrow W^K * T_{emb};\;\; V \leftarrow W^V * T_{emb}$$

where W^K and W^V are learnable weights. The sampled label representations are projected to a (Q)uery matrix

$$Q \leftarrow W^Q * L_r$$

and W^Q is also learnable.

Then we propose an interactive (cross) attention mechanism where the compatibility between the labels (Q) and the inputs (K) are captured in form of an Attention matrix (A). Then the input representations are weighted by the attention weights in A to promote useful dimensions and drown out irrelevant ones.

$$\alpha = \frac{Softmax(Q \cdot K^T)}{\sqrt{d_k}}$$

$$T_{new} \leftarrow \alpha \cdot V$$

where, T_{new} is now the vector representation that fuses the information from content and the representations of the sampled hierarchical labels (Step 8). In the above equation, $Q \in R^{l \times d}$, $K \in R^{n \times d}$ and $V \in R^{n \times d}$. Here l is the number of labels sampled and n is the number of words in the questions (input content). Finally our output embedding from the interactive attention layer $T_{new} \in R^{l \times d}$ as there are l labels sampled during training.

We finally average across the label dimension to compute a fixed length representation for the given question yielding

$$T_{new} = mean(T_{new}, dim = 0)$$

The computed representations are aligned with the corresponding label representations and pulled apart from representations of negative labels using a hinge rank loss function as explained in subsection 3.2.2.

3.2.2 Adaptive hard-negative sampling

In this step, the taxonomy-aware input representations are aligned with the corresponding label representations and pushed apart from the representations of negative labels using a hinge rank loss [107].

For learning representations that disentangle the vector representations of positive and negative labels, we sample hard negatives when optimizing the loss function. The hard negatives are those hierarchical labels with a high semantic relatedness score (cosine similarity) to the input questions but are not the correct hierarchical labels. We sample them using the following equation:

$$hard\_neg \leftarrow top\_k(cos(T_{new}, Index(labels)))$$

where $Index(labels)$ refers to the in-batch hierarchical labels and $label \notin hard\_neg$, $k < batch\_size$. We experiment with different values of k and observe that k =5 gives the best results. After sampling the ground truth hierarchical label as positive and the hard negatives, the hinge rank loss is employed to optimize for the alignment of input and label representations.

The hinge ranking loss is defined as :

$$L(text, l) \leftarrow \sum_{j \neq l} max(0, \delta - cos(T_{emb}, v(l)) + cos(T_{emb}, v(j)))$$

$$L(text, l) \leftarrow \frac{L(text, l)}{len(hard\_neg)}$$

where, $j \in hard\_neg$, T_{emb} denotes the input text embeddings from BERT, $v(label)$ denotes the vector representation of the correct label, $v(j)$ denotes the vector representation

Table 3.2: Comparison of different representation methods for hierarchical labels

Method	Label1 (L1)	Label2 (L2)	cos(L1, L2)
Sentence-BERT	science $\rightarrow$ physics $\rightarrow$ electricity	science $\rightarrow$ chemistry $\rightarrow$ acids	0.3072
Sent2vec	science $\rightarrow$ physics $\rightarrow$ electricity	science $\rightarrow$ chemistry $\rightarrow$ acids	0.6242
GloVe	science $\rightarrow$ physics $\rightarrow$ electricity	science $\rightarrow$ chemistry $\rightarrow$ acids	0.6632

of an incorrect label. The margin value was set to 0.1, which is a fraction of the norm of the embedding vectors (1.0), and resulted in the best performance.

The hard negatives are sampled dynamically during the training and hence are a function of model parameters.

$$hard\_neg \leftarrow f(\theta)$$

where $f(\theta)$ denotes the BERT model and θ denotes the model parameters.

At each iteration in the training loop, we sample the incorrect labels which are closer in the vector space to the input representations. This ensures that the hard negatives improve as the model parameters are updated to better align with the correct label representations. We conduct several ablation studies to compare with random negative sampling and demonstrate that the proposed method aids in high recall retrieval.

3.3 Experiments

3.3.1 Datasets

We collect the following datasets for training the model. Each dataset has learning content like questions or questions with answers tagged to a hierarchical learning taxonomy of form subject$\rightarrow$chapter$\rightarrow$ topic.

QC-Science: We collect this dataset from our partner company Extramarks. There are 47832 question-answer pairs belonging to the science domain tagged with hierarchical labels in this dataset. The dataset consists of 40895 training samples, 2153 validation samples and 4784 test samples. Some samples are shown in Table 3.1. The average number of words per question is 37.14, and per answer, it is 32.01.

ARC(Aristo Reasoning Challenge) [7]: This dataset consists of 7775 science multiple choice exam questions with answer options and 406 hierarchical labels. The average number of words per question is 20.5. The number of samples in the train, validation, and test sets are 5597, 778 and 1400, respectively.

In our experiments, the question and the answer are concatenated and used as the input to the model (BERT), and the hierarchical taxonomy is considered as the label. The number of tokens of each question-answer pair is within 512 and hence can be accommodated within the context limit of BERT.

KhanAcad: We release a new dataset of KhanAcademy video transcripts [2] with corresponding hierarchical labels. This dataset consists of 416 hierarchical labels. The average number of words per question is 822.93. We set maximum length to 512 due to BERT limitations. The number of samples in the train, validation, and test sets are 4188, 924 and 1047,

3.3.2 Analysis of sentence representation methods for representing the hierarchical labels

In this section, we provide an analysis of results from a meta-experiment to decide the best sentence representation methods for the hierarchical labels (learning taxonomy). We embed the hierarchical labels using methods like Sent2Vec [65] and Sentence-BERT [36]. Additionally, we also leverage word embedding methods like GloVe to represent the hierarchical labels by averaging the representations of the tokens in the hierarchical label. We then compute the semantic similarity between the resulting representations of two different hierarchical labels, as shown in 2. From 2, we observe that though "science $\rightarrow$ physics $\rightarrow$ electricity" and "science $\rightarrow$ chemistry $\rightarrow$ acids" are different, a high similarity score is ob-

[2]https://github.com/Khan/khan-api

tained between representations obtained using GloVe embeddings. This may be due to the observation that averaging word vectors can result in loss of information. Additionally, the context of words like physics is not taken into account when encoding the word electricity. Additionally, the words "physics" and "chemistry" are co-hyponyms, which may result in their vectors being close in the continuous vector space when using traditional static embedding methods. We also observe that static sentence embeddings from Sent2Vec are also unable to capture the context of the tokens in the labels, as the representations obtained from Sent2Vec result in a high similarity score. However, we observe that the representations obtained using sentence transformer-based methods like Sentence-BERT are not very similar, as indicated by the similarity score. This indicates that Sentence-BERT is able to produce meaningful sentence representations leveraging the context of tokens for the hierarchical labels. We also observe that Sentence-BERT outputs high similarity scores for semantically related hierarchical labels. We provide a detailed comparison of methods using different vector representation methods as this analysis is not exhaustive.

3.3.3 Methods and Experimental setup

We compare TagRec with flat multi-class classification methods, state-of-the-art multi-label classification methods like HyperIM and other state-of-the-art contrastive learning based methods. In TagRec++, the labels are represented using transformer based sentence representation methods like Sentence-BERT (Sent_BERT) [36] or Universal Sentence Encoder [66].

The methods we compare against are:

- **BERT+Sent2Vec** : In this method, the training and testing phases are similar to TagRec. The label representations are obtained using Sent2vec [65] instead of USE or Sent_BERT.

- **BERT+GloVE** : In this method, the labels are represented as the average of the word embeddings of their constituent words. The word embeddings are obtained from GloVe.

$$V(label) = mean((Gl(subject), Gl(chapter), Gl(topic)))$$

where $V(label)$ denotes vector representation of the label, Gl denotes GloVe pre-trained model. The training and testing phases are the same as TagRec.

- **Twin BERT**: This method is reproduced from the work Twin BERT [108]. In this method, a pre-trained BERT model is fine-tuned to represent the labels in a continuous vector space. The label representations correspond to the first token of the last layer hidden state, denoted as [CLS] in BERT. The two BERT models in the two-tower architecture are fine-tuned simultaneously.

- **BERT multi-class** (label relation) [7]: In this method, the hierarchical labels are flattened and encoded, resulting in a multi-class classification method. Then we fine-tune a pre-trained BERT model for categorizing the input content to the labels. During inference, the representations for the inputs and labels are computed using the fine-tuned model. Then the labels are retrieved and ranked according to the cosine similarity scores computed between the input text representations and the label representations.

- **BERT multi-class** (prototypes) [110]: To provide a fair comparison with *TagRec++*, we propose another baseline that considers the inter-sample similarity than similarity between inputs and labels. A BERT model is fine-tuned like the previous baseline. Then for each class, we compute the mean of the embeddings of random samples from the training set to serve as the prototype for the class. The vector representation for each selected sample is obtained by the concatenation of the [CLS] token obtained from the last 4 layers of the fine-tuned model. This method of vector representation gives the best performance. After obtaining the class prototypes, at inference, we

Table 3.3: Performance comparison of TagRec++ using Recall@k (R@k) and Mean Reciprocal Rank@k (MRR@k) with variants and baselines, † indicates TagRec++'s significant improvement at 0.001 level using *t-test*

Dataset	Method	R@1	R@3	R@5	MRR@1	MRR@3	MRR@5
	TagRec++(BERT+USE) (ours)	**0.65** †	**0.84** †	**0.89** †	**0.65** †	**0.74** †	**0.75** †
	TagRec++(BERT+SB) (ours)	**0.65** †	**0.85** †	**0.90** †	**0.65** †	**0.75** †	**0.76** †
	TagRec(BERT+USE) [109]	0.54	0.78	0.86	0.54	0.65	0.67
	TagRec(BERT+SB) [109]	0.53	0.77	0.85	0.53	0.64	0.66
QC-Science	BERT+sent2vec	0.43	0.70	0.79	0.43	0.56	0.58
	Twin BERT [108]	0.32	0.60	0.72	0.32	0.44	0.47
	BERT+GloVe	0.39	0.65	0.76	0.39	0.50	0.53
	BERT classification (label relation) [7]	0.19	0.33	0.39	0.19	0.25	0.27
	BERT classification (prototypes) [110]	0.54	0.75	0.83	0.54	0.63	0.65
	Pretrained Sent_BERT	0.11	0.22	0.30	0.11	0.16	0.18
	HyperIM [111]	0.57	0.79	0.85	0.57	0.33	0.21
	TagRec++(BERT+USE) (ours)	**0.48** †	**0.66** †	**0.75** †	**0.48** †	**0.56** †	**0.58** †
	TagRec++(BERT+SB) (ours)	**0.49** †	**0.71** †	**0.78** †	**0.49** †	**0.59** †	**0.61** †
	TagRec(BERT+USE) [109]	0.35	0.55	0.67	0.35	0.44	0.47
	TagRec(BERT+SB) [109]	0.36	0.55	0.65	0.36	0.44	0.46
ARC	BERT+sent2vec	0.22	0.43	0.55	0.22	0.28	0.31
	Twin BERT [108]	0.14	0.31	0.46	0.14	0.21	0.24
	BERT+GloVe	0.23	0.43	0.56	0.23	0.32	0.35
	BERT classification (label relation) [7]	0.11	0.21	0.27	0.11	0.15	0.16
	BERT classification (prototypes) [110]	0.35	0.54	0.64	0.35	0.43	0.45
	Pretrained Sent_BERT	0.12	0.24	0.31	0.12	0.17	0.19
	HyperIM [111]	0.20	0.34	0.40	0.20	0.17	0.12
	TagRec++(BERT+USE) (ours)	**0.37** †	**0.50** †	**0.55** †	**0.37** †	**0.43** †	**0.44** †
	TagRec++(BERT+SB) (ours)	**0.38** †	**0.54** †	**0.61** †	**0.38** †	**0.45** †	**0.46** †
	TagRec(BERT+USE) [109]	0.30	0.45	0.50	0.30	0.37	0.38
	TagRec(BERT+SB) [109]	0.26	0.44	0.52	0.26	0.34	0.36
KhanAcad	BERT+sent2vec	0.16	0.32	0.42	0.16	0.23	0.25
	Twin BERT [108]	0.10	0.22	0.32	0.10	0.15	0.17
	BERT+GloVe	0.16	0.31	0.41	0.16	0.22	0.25
	BERT classification (label relation) [7]	0.06	0.14	0.20	0.06	0.09	0.11
	BERT classification (prototypes) [110]	0.18	0.30	0.35	0.18	0.23	0.24
	Pretrained Sent_BERT	0.06	0.11	0.15	0.05	0.08	0.09
	HyperIM [111]	0.19	0.29	0.34	0.19	0.13	0.10

obtain the embeddings for each test sample and compute cosine similarity with the class prototype embeddings. Then the top-k classes ranked as per cosine similarity are returned.

- **Pretrained Sent_BERT**: We implement a baseline where the input texts and labels are encoded using a pre-trained Sentence-BERT model. For each input, the top-k labels are retrieved by computing cosine similarity between the input and the label representations.

- **TagRec** [109]: This can be considered as ablation or preliminary version of TagRec++, where the attention mechanism and adaptive hard negative sampling are the missing modules when compared to TagRec++.

- **HyperIM [111]**: We also compare with the recent SOTA approach HyperIM [111] which casts the input and the hierarchical label to the hyperbolic space. Then the distance between the input and all label representations are used as a feature vector for classification. This approach cannot adapt to changes in label space.

Some other ablations we perform are:

- **TagRec++ (-attention)**: We perform an experiment with the removal of the interactive attention mechanism explained in section 3.2 from TagRec++ and compare it with the original approach to test the effectiveness of the proposed attention mechanism. We perform this experiment for both variants of TagRec++.

- **TagRec++ (-hard-negatives)**: In this variant of TagRec++, instead of the adaptive in-batch hard negative sampling, we replace them with adaptive random negatives. We sample the same number of negatives for a fair comparison. We perform this experiment to analyze the impact of our adaptive hard-negative sampling approach on the final performance of the proposed approach.

3.3.4 Metrics

We compare the proposed approaches with baselines and state-of-the-art methods using standard IR metrics like Recall@k (R@k) and MRR@k [112, 113] which are popular for retrieval tasks with only one relevant label. Since each sample in our datasets are tagged only with one relevant hierarchical path, R@k helps evaluate if the correct path is among the top-k retrieved learning taxonomy paths. Additionally, Mean Reciprocal Rank (MRR) [114] helps measure if the relevant label is assigned a higher rank among retrieved labels. This is of paramount importance, as the higher the rank of the relevant label, the higher the retrieval quality. Since we have only one relevant label, MRR is also equivalent to Mean Average Precision [114]. We consider R@k instead of just R@1 as academicians may be interested in other relevant paths for content categorization. Though the considered datasets have only one hierarchical taxonomy path as ground truth, in reality a learning content could be categorized to multiple relevant taxonomy paths.

3.4 Results and Analysis

The performance comparison of TagRec++ with baselines and other variants can be observed from Table 3.3. We observe that TagRec++ outperforms all approaches as measured by Recall@k (R@k).

3.4.1 Comparison with other approaches

We observe that contextualized embedding *TagRec++* provides the best performance when compared to state-of-the-art contrastive learning and multi-label, multi-class classification approaches. Further, TagRec++(BERT+USE) and TagRec++(BERT+Sent_BERT) outperform approaches which leverage static sentence embedding methods like BERT+Sent2Vec and BERT+GloVe. This is because transformer-based encoding methods use self-attention to produce better representations. In addition, the Sentence-BERT and the Universal Sentence Encoder models are ideal for retrieval based tasks as they were pre-trained on semantic text similarity (STS) tasks.

TagRec++ outperforms the TwinBERT architecture, which is a bi-encoder model that simultaneously learns input and label representations. However, it does not model the interaction between input and label representations. Additionally, the fine-tuning of two BERT models is computationally expensive and converges later than the proposed approach. Since the [CLS] token from BERT is taken as the final representation, it does not also explicitly capture the compositional relationship between terms in the hierarchical labels [67].

TagRec++ also outperforms state-of-the-art multi-class classification approach, Hy-

Table 3.4: Examples demonstrating the performance for unseen labels at test time.

Question text	Ground truth	Top 2 predictions	Method
A boy can see his face when he looks into a calm pond. Which physical property of the pond makes this happen? (A) flexibility (B) reflectiveness (C) temperature (D) volume	matter→properties of material→reflect	matter→properties of material→flex and **matter→properties of material→reflect**	TagRec++ (BERT+USE)
		matter→properties of objects→mass and matter→properties of objects→density	Twin BERT [108]
		matter→states→solid and matter→properties of material→density	BERT+GloVe
		matter→properties of material→specific heat and matter→properties of material	BERT+sent2vec
Which object best reflects light? (A) gray door (B) white floor (C) black sweater (D) brown carpet	matter→ properties of material→reflect	energy→light→reflect and **matter→properties of material→reflect**	TagRec++ (BERT+USE)
		energy→thermal→ radiation and energy→light→generic properties	Twin BERT [108]
		energy→light and energy→light→refract	BERT+GloVe
		energy→light→reflect and energy→light→refract	BERT+sent2vec

perIM. HyperIM models the interaction between the input and the labels in the hyperbolic space and uses the resulting representation for classification. It is notable that it cannot adapt to changes in the label space, owing to the final linear layer. We also observe that the relevant labels are ranked lower by HyperIM compared to our approach as measured by MRR@k.

Finally, we observe that the TagRec++ method outperforms the flat multi-class classification based baselines, confirming the hypothesis that capturing the semantic relatedness between the terms in the input and tokens in the hierarchical labels results in better representations. This is pivotal to the question-answer pair categorization task, as the technical terms in the short input text are semantically related to the tokens in the label. The baseline (BERT label relation) performs poorly, as it has not been explicitly trained to align the input and the hierarchical label representations. The representations obtained through the flat multi-class classification approach have no notion of semantic relatedness between the content and label representations. But the prototypical embeddings baseline performs better, as the classification is done based on semantic matching between train and test sample representations. However, this baseline also has no notion of semantic relatedness between the input and label representations. Hence, it does not perform well when compared to our proposed method, TagRec++. Moreover, this baseline cannot also adapt to changes in the label space and requires a change in the final classification layer and *retraining*. We also observe that the baseline of semantic matching using pre-trained sentence BERT does not work well.

We also observe that *TagRec++* provides a higher rank to relevant labels, as indicated by MRR@k. This is important for real-world deployments to reduce the cognitive load for academicians when onboarding new content.

To confirm the efficacy of *TagRec++*, we perform statistical significance tests and observe that the predicted results are statistically significant over *TagRec*. For instance, for Recall@5 we observe that the predicted outputs from TagRec++ are statistically significant (*t-test*) with p-values **0.0000499** and **0.0000244** for *QC-Science* and *ARC* respectively.

3.4.2 Performance on unseen labels:

The TagRec++ was also able to adapt to changes in the label space. For instance, in the *ARC* dataset, two samples in the test set were tagged with *"matter→properties of material→reflect"* unseen during the training phase as shown in Table 3.4. At test time,

Table 3.5: Ablation analysis of TagRec++

Dataset	Method	R@1	R@3	R@5	MRR@1	MRR@3	MRR@5
QC-Science	TagRec++(BERT+USE) (proposed method)	**0.65**	**0.84**	**0.89**	**0.65**	**0.74**	**0.75**
	TagRec++(BERT+SB) (proposed method)	**0.65**	**0.85**	**0.90**	**0.65**	**0.75**	**0.76**
	TagRec++(BERT+USE) (- attention)	0.62	0.83	0.88	0.62	0.69	0.70
	TagRec++(BERT+SB) (- attention)	0.62	0.83	0.87	0.62	0.70	0.71
	TagRec++(BERT+USE) (- hard-negatives)	0.57	0.81	0.86	0.57	0.69	0.70
	TagRec++(BERT+SB) (- hard-negatives)	0.56	0.80	0.87	0.56	0.64	0.66
ARC	TagRec++(BERT+USE) (proposed method)	**0.48**	**0.69**	**0.75**	**0.48**	**0.56**	**0.58**
	TagRec++(BERT+SB) (proposed method)	**0.49**	**0.71**	**0.77**	**0.48**	**0.59**	**0.61**
	TagRec++(BERT+USE) (- attention)	0.41	0.61	0.70	0.41	0.47	0.49
	TagRec++(BERT+SB) (- attention)	0.44	0.64	0.74	0.44	0.52	0.54
	TagRec++(BERT+USE) (- hard-negatives)	0.39	0.60	0.72	0.39	0.48	0.51
	TagRec++(BERT+SB) (- hard-negatives)	0.45	0.66	0.74	0.45	0.54	0.56
KhanAcad	TagRec++(BERT+USE) (ours)	**0.37**	**0.50**	**0.55**	**0.37**	**0.43**	**0.44**
	TagRec++(BERT+SB) (ours)	**0.38**	**0.54**	**0.61**	**0.38**	**0.45**	**0.46**
	TagRec++(BERT+USE) (- attention)	0.33	0.49	0.53	0.33	0.40	0.41
	TagRec++(BERT+SB) (- attention)	0.26	0.44	0.53	0.26	0.34	0.36
	TagRec++(BERT+USE) (- hard-negatives)	0.32	0.48	0.53	0.32	0.39	0.41
	TagRec++(BERT+SB) (- hard-negatives)	0.31	0.48	0.57	0.31	0.39	0.41

the label *"matter→properties of material→reflect"* appeared in top 2 predictions output by the proposed method (TagRec++ (BERT + USE)) for the two samples. We also observe that for the method (TagRec++ (BERT + Sent_BERT)) the label *"matter→properties of material→reflect"* appears in its top 5 predictions. We observe that for other baselines shown in Table 3.4 the correct label does not get retrieved even in top-10 results. The top 2 results retrieved from other methods for the samples are shown in Table 3.4. Similar results can be observed for the BERT classification (label relation) and BERT classification (prototypes) baselines. We do not show them in Table 3.4 owing to space constraints. The top 2 predictions from BERT classification (prototypes) baseline for example 1 in Table 3.4 are *matter→properties of objects→temperature* and *matter→properties of objects→shape*.

For example 2, in Table 3.4, the top 2 predictions from BERT classification (prototypes) are *energy→light→reflect* and *matter→properties of material→color*.

The top 2 predictions from BERT classification (label relation) baseline for example 1 in Table 3.4 are *matter→properties of objects→ density* and *matter→ properties of material→density*. For example 2, in Table 3.4, the top 2 predictions from BERT classification (label relation) are *energy→light→refract* and *matter→properties of material→luster*. This validates our hypothesis that the proposed method can adapt to new

Table 3.6: Performance comparison for zero-shot learning objective categorization

Method	R@1	R@2
TagRec++(BERT+SB) (ours)	**0.82**	**0.94**
TagRec++(BERT+USE) (ours)	0.79	0.93
TagRec(BERT+USE)	0.69	0.85
TagRec(BERT+SB)	0.77	0.91
BERT+sent2vec	0.49	0.64
Twin BERT [108]	0.54	0.79
BERT+GloVe	0.62	0.84
BERT classification (label relation) [7]	0.46	0.59
BERT classification (prototypes) [110]	0.60	0.76
Pretrained Sent_BERT	0.39	0.54

labels without changes in model architecture and retraining, unlike existing methods.

3.4.3 Ablation studies:

We also perform several ablation analyses of the proposed TagRec++ approach. As observed in Table 3.5 we compare TagRec++ with and without the proposed interactive attention mechanism. We see a clear performance difference, confirming the hypothesis that the interactive attention mechanism is crucial for high recall retrieval, as it captures the relatedness between the tokens in the hierarchical labels and the terms in the input content (questions).

We also performed another ablation study to ascertain the effectiveness of the proposed in-batch hard negative sampling. Instead of sampling in-batch hard negatives, we sample random negatives. The random negatives are also sampled dynamically for a fair comparison. From Table 3.5, we can observe that the proposed in-batch hard negative sampling works better than the random negative sampling. In the hard negative sampling approach, as the model is trained to align the content and appropriate label representations, it gets better at sampling hard negatives which result in high-recall retrieval.

Table 3.7: Ablation results for zero-shot evaluation on learning objective categorization

Method	R@1	R@2
TagRec++(BERT+SB) (ours)	**0.82**	0.94
TagRec++(BERT+SB) (-attention)	0.80	0.94
TagRec++(BERT+SB) (-hard-negatives)	0.80	0.93

Table 3.8: Performance comparison for each epoch on ARC dataset: TagRec++ vs TagRec++ (-hard-negatives)

	Epochs				
Method	**2**	4	6	8	10
TagRec++(BERT+SB) (ours)	**0.30**	**0.40**	**0.43**	**0.45**	**0.45**
TagRec++(BERT+SB) (-hard-negatives)	0.22	0.33	0.35	0.39	0.39

3.4.4 Zero-shot performance:

We curated a set of learning objectives from K-12 textbooks to test the ability of *TagRec++* to tag related short learning content without training. This experiment is performed to observe the zero-shot abilities of *TagRec++*. We observe that *TagRec++* outperforms existing approaches as measured by Recall@k as shown in Table 3.6. This demonstrates that the proposed approach also leads to high recall retrieval in a zero-shot setting.

We also perform certain ablation studies for the zero-shot setting by removing the interactive attention component and the in-batch hard negatives sampling approach as shown in Table 3.7. We observe that *TagRec++* achieves the highest performance, indicating the significance of the proposed attention mechanism and hard negative sampling method.

3.4.5 Epochwise comparison of hard negatives vs random negatives:

The in-batch dynamic negative sampling is based on the hypothesis that the model improves with training and samples better hard negatives, leading to high recall retrieval when compared to dynamic random negatives sampling. To test this hypothesis, apart from the ablation shown in Table 3.5, we also perform an epochwise comparison of dynamic hard

Table 3.9: Qualitative analysis of top-3 hard-negatives sampled on QC-Science dataset

Question	Answer	Epoch	**Hard negatives**		
			1	2	3
In the given transistor circuit, the base current is 35 μA. The value of R b is	200 Omega	1	science $\rightarrow$ physics $\rightarrow$ work, energy and power	physics $\rightarrow$ part - ii $\rightarrow$ mechanical properties of fluids	physics $\rightarrow$ part - ii $\rightarrow$ thermal properties of matter
		5	science $\rightarrow$ physics $\rightarrow$ communication systems	physics $\rightarrow$ part - i $\rightarrow$ magnetism and matter	physics $\rightarrow$ part - i $\rightarrow$ system of particles and rotational motion
		10	physics $\rightarrow$ part i $\rightarrow$ magnetic effects of current	physics $\rightarrow$ part - i $\rightarrow$ magnetism and matter	physics $\rightarrow$ part - i $\rightarrow$ moving charges and magnetism

negatives and dynamic random negatives as shown in Table 3.8. We observe that *TagRec++* with dynamic hard negatives has a higher recall in each epoch due to better sampling when compared to dynamic random negatives. This demonstrates that hard negatives lead to better recall than random negatives. Also, the proposed dynamic sampling approach leads to better hard negatives as the model learns to align the content and hierarchical label representation sub-spaces.

3.4.6 Qualitative analysis

We analyze the hard-negatives sampled in the training loop to determine if the quality of hard negatives increases as training progresses. The observation for a sample is shown in Table 3.9. We observe in epoch one, the hard negatives are centered around *physics* subject, but the topics are not related to the input. The ground truth label for the question shown in the is centered around *electrical circuits*. We observe that as training progresses, the top-2 labels are centered around *magnetism*, *communication systems* but do not correspond to the correct theme of *electrical circuits*, rendering them as hard negatives. This demonstrates that dynamic sampling of in-batch hard-negatives improves with training. We observe a similar phenomenon for other samples too, which are not attached here due to space

Table 3.10: Performance comparison for different values of k when sampling top-k taxonomy tags for attention in TagRec++

# of tags for attention	R@1	R@3	R@5
1	0.61	0.82	0.88
5	**0.65**	**0.85**	**0.90**
10	0.63	0.85	0.89

constraints.

We vary the value of k for the top-k tags (hierarchical labels) sampled for cross-attention, discussed in Section 3.2. The results are shown in Table 3.10. We observe that the highest R@k is achieved for value of 5. When only one tag is sampled, it contains very less information, as demonstrated by the values of R@k. We also observe sampling ten tags to fuse the taxonomy information with the input leads to a lower R@k. This maybe due to noise induced by the tags less related to the input. Hence, we set k in top-k tags sampled to 5.

We also perform the qualitative analysis of tags sampled as shown in Table 3.11. We observe that though in the first epoch, we get tags related to *magnetism* as the training progresses, in later epochs, we get most tags relevant to *electricity* which are closer to the ground truth label *semiconductors and electrical circuits*. This helps the model capture the information in tokens from various tags relevant to the topic and the terms in the input question.

3.4.7 Execution Time

We observe that inference is faster for the proposed approach as the embeddings for labels are pre-computed and indexed. We observe that for QC-Science for a test set of 4784 questions the inference time on T5 GPU without batched inference is **3 minute 20 seconds**. We posit it would be faster on V100 GPUs and batched setting. We observe that the proposed *TagRec++* converges faster than other competitive methods owing to the proposed cross-interaction approach. For instance, on QC-Science we observe *TagRec++ (BERT+SB)* converges after 20 epochs leading to training time of **252 minutes (14 minutes per epoch)** and *TagRec (BERT+SB)* has training time of **420 minutes (30 epochs)**. Twin BERT on QC-science has a training time of **930 minutes** with 31 minutes per epoch due to fine-tuning of two BERT models.

3.4.8 Insights

- *TagRec++* outperforms existing state-of-the-art contrastive learning approaches and multi-label, multi-class classification approaches.
- *TagRec++* provides impressive R@1 performance. However, in real world deployments, more than one taxonomy path is applicable for each learning content. Hence, we also report R@3 and R@5 as done in traditional information retrieval setting [112]. We observe that with increasing k relevant documents are ranked higher compared to other methods as measured by MRR. Additionally, retrieving top-k taxonomy paths also helps in assisting the academicians to select the appropriate taxonomy path without inferring significant cognitive load.
- *Tagrec++* adapts to changes in the label space and demonstrates good zero-shot performance in tagging related content.

3.5 Discussions

We proposed a novel approach, TagRec++ for tagging content to hierarchical taxonomy. The proposed approach can be used in online systems to onboard and tag content to standardized taxonomy. We proposed an adaptive in-batch hard-negative sampling approach for achieving high recall retrieval. We also propose a cross-attention approach where we fuse the information from the content and the hierarchical label embeddings to capture the interaction between the tokens in the hierarchical labels and the terms in the input content.

Table 3.11: Qualitative analysis of top-3 tags sampled on QC-Science dataset for the cross-attention mechanism

			hierarchical tags for attention		
Question	Answer	Epoch	**1**	2	3
In the given transistor circuit, the base current is 35 μA. The value of R b is	200 Omega	1	physics $\rightarrow$ part - i $\rightarrow$ magnetism and matter	physics $\rightarrow$ part - i $\rightarrow$ moving charges and magnetism	science $\rightarrow$physics$\rightarrow$magnetic effects of electric current
		5	physics $\rightarrow$ part - i $\rightarrow$ electromagnetic waves	physics $\rightarrow$ part - i $\rightarrow$ alternating current	physics $\rightarrow$ part - i $\rightarrow$ electrostatic potential and capacitance
		10	physics $\rightarrow$ part - i $\rightarrow$ alternating current	physics $\rightarrow$ part - i $\rightarrow$ current electricity	physics $\rightarrow$ part - i $\rightarrow$ electric current and it's effects

We observe that the proposed approach outperforms *TagRec*, other baselines and achieves high recall retrieval.

However, a serious limitation and requirement of this approach is that the taxonomy must have semantic context and must be words grounded in a knowledge base. Applications using serial numbers or other conventions in taxonomy would not be able to leverage *TagRec*++ for high recall retrieval.

Since, the taxonomy is of hierarchical nature, we also plan to explore the hyperbolic space for projecting the content and taxonomy representations. In the future, we plan to explore the projection of the proposed efficient cross-attention mechanism to the hyperbolic space.

The proposed approach, *TagRec*++, can act as a fundamental tool for the organization of content according to a taxonomy. This is evident from the usage of the approach in *QDup* [42], a large-scale system for detecting duplicate questions in large data repositories. Since pairwise comparisons of all questions are computationally heavy, the authors leverage *TagRec++* in a zero-shot setting to tag the questions to a standard taxonomy. Once tagged, the taxonomy labels act as buckets. An incoming new question is tagged to a taxonomy, which is used as a key to index to the appropriate bucket. This question is only compared to questions that fall under the same taxonomy. This work further demonstrates the zero-shot capabilities of the proposed approach and its ability to adapt to the changes in the label space.

We also leverage *TagRec++* for building a tool to generate learning objectives. We link repositories of learning content and existing learning objectives using *TagRec++*. We detail the approach in the upcoming chapters.

In this work, the content is tagged to an existing taxonomy. The depth of the taxonomy is limited to the topic or sub-topics collected. However, in certain cases, it is desirable further to expand the taxonomy for fine-grained tagging and faceted retrieval. For instance, a fine-grained latent concept like "The Archimedes principle" is pivotal to solving a question related to the density of objects immersed in water. Discovering such concepts can also help in linking related content and tailored content recommendations that are useful for the learner. To this end, in the next chapter, we propose a novel unsupervised approach for keyphrase extraction to aid in fine-grained concept discovery.

CHAPTER 4
FINE-GRAINED CONCEPT EXTRACTION USING TOPIC AWARE CONTEXTUALIZED REPRESENTATIONS

4.1 Introduction

Concepts are keyphrases that reflect the core ideas of a document. They are brief and are less frequent than other noun phrases. They play an important role in many text processing applications like document clustering, classification, information retrieval [115, 116], and text generation. We map the task of concept extraction to the well-known task of Automatic Keyphrase Extraction (AKE) [1]. In addition to the above-mentioned applications, we are primarily interested in the applications of AKE in online learning platforms.

Learning contents in online learning platforms are tagged at the topic level for accessibility. However, a topic can be further divided into concepts that enable the linking of related learning content and easy navigation through the learning content. In this chapter, concepts are characterized as keyphrases as they describe the content of a document. We posit that the Automatic Keyphrase Extraction (AKE) from learning content can help to index the massive collection of learning content in online learning platforms enabling better accessibility of the learning content. Automatic Keyphrase Extraction is a well-studied problem [30]. Unlike the supervised methods, the unsupervised methods do not require annotated documents and rely on in-corpus statistical information for extracting keyphrases. In most of the unsupervised keyphrase extraction methods [31, 32, 33, 34] the candidate phrases are represented by a word graph formed based on a co-occurrence window and then ranked. Recent unsupervised methods like EmbedRank [35] leverage representation learning methods that help to capture the semantic relatedness between the phrases and the

[1]This chapter is partly a reproduction of a paper published at European Conference on Information Retrieval (ECIR) [117]

document.

We propose a novel unsupervised method to automatically extract keyphrases from a given document. In this method, the candidate phrases and the given document are represented in a continuous vector space by combining the contextual embeddings and the topical information from LDA [73] to strengthen the associations between phrases that occur in similar contexts and also represent similar topics. Then a graph-based ranking algorithm where the nodes are represented by phrases rather than words is employed to rank the phrases. The proposed unsupervised method helps to capture two important characteristics needed for keyphrases: *coherence* and *informativeness*. We posit that the selected phrases are *coherent* if they convey a consistent idea [37] and they are *informative* if they convey the core ideas discussed in the document. In the proposed method, *coherence* is captured as the cosine similarity computed between the embeddings of the candidate phrases. The *informativeness* of a phrase is captured as the cosine similarity computed between the embeddings of the candidate phrase and the document. The proposed algorithm outperforms existing unsupervised AKE methods. For instance, on SemEval2017 dataset, our method advances the F1 score from 0.2195 (EmbedRank) to **0.2819**.

Following are the core technical contributions of our paper:

- We propose a new topic-aware representation method to represent the phrases and the document. To the best of our knowledge, this representation method has not been applied to the task of keyphrase extraction.
- We propose a graph-based ranking method with a new scoring mechanism that captures the *informativeness* and *coherence* measures by using the proposed representation method.
- We apply our algorithm on the task of enriching the set of extracted keyphrases with new keyphrases which are not present in the source document by using external knowledge sources like Wikipedia.

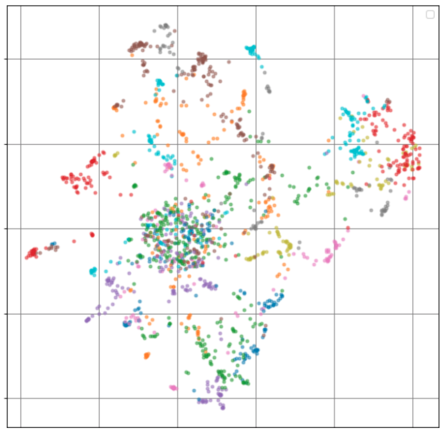
The code and data can be found at https://github.com/VenkteshV/Unsupervised_keyphrase_extraction_CoTagRank_ECIR_2022.

4.2 Methodology: An Unsupervised Topic Aware Keyphrase Extraction Approach

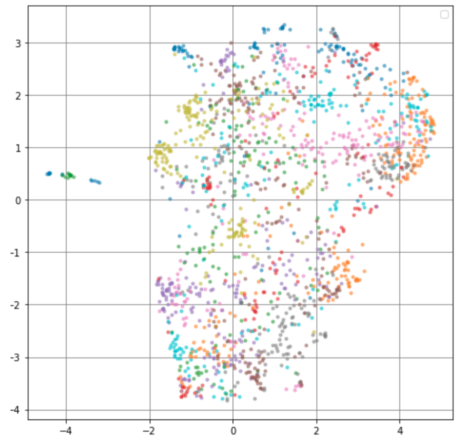
In this section, we describe the proposed extraction algorithm, CoTagRank. First, the candidate phrases are extracted based on the Part Of Speech (POS) tags using the pattern $< NN.*|JJ > * < NN.* >$ [69]. Then the phrases and the document are projected to a continuous vector space and phrases are ranked as discussed in the following sections.

4.2.1 Vector representations for the phrases and the document

The primary goal of our algorithm is to extract the candidate phrases that best describe the document. We define two measures for achieving this goal, namely *coherence* and *informativeness*. The coherence measure can be seen as an indicator that the candidate phrases represent a consistent idea. The informativeness measure can be seen as an indicator of whether the phrases convey the core ideas discussed in the document. We posit that the above two measures can be captured by leveraging the topical information. Hence, we give a novel vector representation mechanism that combines topical information with the embeddings obtained from the state-of-the-art contextualized embedding methods. We leverage contextualized embeddings to handle polysemous words. An example of polysemy can be seen in the following two sentences: “Consider an **imaginary** box”, “An **imaginary number** is a complex number”. In the above two sentences, the word “imaginary” has different meanings. Contextualized embeddings capture the context of usage of the word and hence produce different vector representations for the same word depending on the context. However, we need a proper analysis to determine the suitable representation method that can capture the mentioned properties without loss of semantic relatedness.



(a) TF-IDF result



(b) BERT result

Figure 4.1: Visualization of clustering results on vectors from TF-IDF and BERT

Meta-experiment to Determine the Best Representation Method

We conducted several experiments to compare various embedding methods on the task of clustering the vector representation of the documents (lecture transcripts) to determine the best representation method. The clustering task was performed only to determine the best representation method and is not directly used for concept extraction. We encode the documents to a continuous vector space using TF-IDF [2] and Sentence-BERT [3] to obtain sentence embeddings for the documents. For this series of experiments, we used the data we extracted from khan academy. We crawled 2400 video transcripts for the science domain.

Then we cluster the vector representations using K-Means with the number of clusters set to 50 and visualize the clustering result on vectors from TF-IDF and Sentence-BERT using UMAP [118].

From Figure 2a and 2b, we observe that TF-IDF does not produce separated or well-balanced clusters. On the other hand, BERT produces better clusters, but still, the clusters are not well separated.

For a proper quantitative analysis, we compute **Silhouette coefficient** to interpret and

[2]https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html
[3]https://github.com/UKPLab/sentence-transformers

Method	**Silhouette Coefficient**
USE+LDA	**0.45**
Sentence-BERT+LDA	0.144
TF-IDF	0.025
BERT	0.032

Table 4.1: Silhouette coefficient of various vector representation methods.

validate consistency within clusters of data.

$$SilhouetteCoefficient = \frac{(a-b)}{max(a,b)}$$

where **b** is the mean intra-cluster distance, **a** depicts inter-cluster distance i.e. mean distance to the instances of the immediate closest cluster A value of 1 indicates instance is assigned to correct cluster and close to 0 or -1 indicates an incorrect cluster.

We observed only a minor difference in silhouette score between BERT and TF-IDF.

Then we compared the above representations to our novel representation that leverages both topic information from LDA (Latent Dirichlet Allocation) [73] and sentence embeddings from BERT and Universal Sentence Encoder (USE). LDA is a generative probabilistic model in which each word in document d is assumed to be generated by sampling a topic from d's topic distribution θ^d and then sampling a word from the distribution over words denoted by ϕ^t of a topic.

4.2.2 Proposed Vector Representation Method

The phrase representations are obtained by combining the contextualized embeddings of the phrases with the topic vectors of their constituent words obtained from LDA. The LDA is a generative probabilistic model in which each word in document d is assumed to be generated by sampling a topic from *d*'s topic distribution θ^d and then sampling a word from the distribution over words denoted by ϕ^t of a topic. We use a pre-trained Universal Sentence Encoder (USE) to obtain contextualized embeddings for both the phrases and the sentences as it has been pre-trained on the Semantic Text Similarity (STS) task. This representation method helps in bringing the phrases that are semantically related and having similar topic distributions closer in the vector space. This implies that the phrases that are both semantically related and represent similar topics would have a higher coherence measure (cosine similarity between phrase representations). The phrase representations are obtained in the following manner:

$$LE(CP) = \sum_{w \in CP} [p(w|t_1), p(w|t_2)...]$$

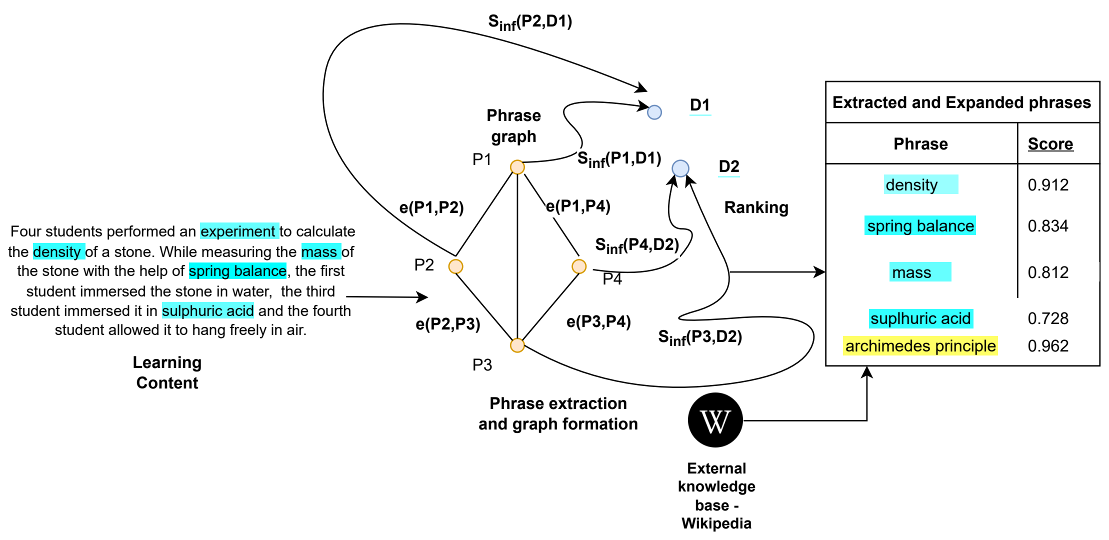
$$CPE = concat(LE(CP), CE(CP)) \tag{4.1}$$

where LE represents LDA embeddings, CP represents a candidate phrase, CE represents contextualized embeddings, and CPE represents candidate phrase embeddings. The vector $[p(w|t_1), p(w|t_2)...]$ represents the word-topic probabilities that are derived from the word distributions ϕ^t over the topics. Similarly, the document representation can be obtained by combining the topic distribution of the document with the contextualized embeddings of the document sentences. The document representation is obtained as follows:

$$LE(doc) = [p(t_1|d), p(t_2|d)...]$$

$$DE = concat(LE(doc), CE(doc)) \tag{4.2}$$

where, the vector $[p(t_1|d), p(t_2|d)...]$ represents the document-topic probabilities and DE represents the document embedding. Latent Dirichlet Allocation (LDA) is run only once on the corpus of documents and not for every document. The vector representations obtained as described are leveraged in the graph-based ranking step to compute final scores for the candidate phrases.



Four students performed an experiment to calculate the density of a stone. While measuring the mass of the stone with the help of spring balance, the first student immersed the stone in water, the third student immersed it in sulphuric acid and the fourth student allowed it to hang freely in air.

Extracted and Expanded phrases

Phrase	Score
density	0.912
spring balance	0.834
mass	0.812
suplhuric acid	0.728
archimedes principle	0.962

Figure 4.2: Proposed approach for Phrase Extraction and Expansion

4.2.3 Graph based ranking

In this section, we discuss in detail the proposed ranking approach to extract high-quality phrases. The workflow of the proposed extraction algorithm is shown in Figure 4.2.

It differs from traditional methods like TextRank, SingleRank and PositionRank. We construct an undirected graph with the candidate phrases as the vertices instead of words. Constructing the graph in this manner circumvents the overgeneration errors that occur when the phrases are ranked by aggregating the scores of the top-ranked words. Hence using word graph-based AKE methods may result in an uninformative phrase being assigned a high score just because one of the constituent words has a higher score. The edges connecting the phrases are weighted by the semantic relatedness (cosine similarity) computed between the vector representations of the phrases. The vector representations for the phrases are obtained as described in the previous subsection. The edges are formed between the phrases (nodes) that co-occur in the original text within a specified window size (tunable parameter). We demonstrate that when the window size is set to the maximum value for forming a complete graph, we get the maximum performance. The completeness nature of the graph has the benefit of connecting phrases together that may not co-occur together but have similar topic distributions. The complete graph also provides good convergence guarantees. The random walk on a complete graph is equivalent to an irreducible Markov chain where every state is reachable from every other state, which ensures convergence to a finite stationary distribution π.

As mentioned in the previous subsection, our goal is to rank those phrases higher that are coherent and informative for the document. The *coherence* measure is represented by the edge weights of the graph. The *informativeness* measure for each phrase (P_a) is the normalized cosine similarity between the document and the phrase representations computed as follows:

$$n\_Sim(P_a, doc) = \frac{Sim(P_a, doc) - min_{Pb \in P}(Sim(P_b, doc))}{max_{P_b \in P}(Sim(P_b, doc))} \tag{4.3}$$

where n_Sim is normalized cosine similarity, Sim is the cosine similarity function and doc represents the document. Then the similarity metric is obtained as :

$$F\_Sim(P_a, doc) = \frac{n\_Sim(P_a, doc) - \mu(n\_Sim(P, doc))}{\sigma(n\_Sim(P, doc))} \tag{4.4}$$

where P is the set of phrases. The F_Sim function returns the final cosine similarity metric obtained after normalization and standardization of cosine similarities. The function n_Sim given a set of embeddings of the phrases and a document embedding as inputs return a vector of normalized cosine similarities. Each element in the output vector is the normalized similarity between the corresponding embedding in the set and the document embedding. In Equation (4.3), $n\_Sim(Pa, doc)$, P_a denotes a set of embeddings having only one element. Whereas in Equation (4), $n\_Sim(P, doc)$, P has multiple embeddings.

The goal is to find the phrases that maximize the objective:

$$Obj = \lambda S_{coh}(P) + (1 - \lambda) S_{inf}(P, doc)$$

Table 4.2: Statistics of the datasets used

Dataset	Domain	# of docs	# of tokens/-doc	# of gold keys	# of gold keys/-doc
Inspec	Science	2000	128.20	29230	14.62
SemEval2017	Science	493	178.22	8969	18.19
SemEval2010	Science	243	8332.34	4002	16.47

where, S_{inf} denotes the function that returns the *informativeness* measure computed using Equation 4.4.

The S_{coh} is the function that computes the *coherence* measure. The S_{coh} denotes a function that takes a set of embeddings of phrases (P) and outputs a vector of cosine similarities computed between embeddings of all possible pairs of distinct phrases in the set. The parameter λ balances the importance given to S_{coh} and S_{inf} factors.

Iteratively optimizing the above objective is similar to random walk based approaches. Hence, maximizing the above objective can be done as follows:

Every candidate phrase in the graph is ranked by:

$$R(p_i) = \lambda \sum_{j:p_j->p_i} \frac{e(p_i,p_j)}{OutDeg(p_j)} R(p_j) + (1-\lambda) S_{inf}(p_i) \tag{4.5}$$

where $e(p_i, p_j)$ denotes the weight of the edges between the phrases (p_i and p_j) (*coherence*) and $S_{inf}(p_i)$ is the *informativeness* score that helps in biasing the random jump to phrases (vertices) that are closer to the document in the vector space.

4.3 Experiments

4.3.1 Datasets

We evaluate our algorithm on standard datasets like Inspec [119], SemEval 2017 [120] and SemEval 2010 [121] for keyphrase extraction. We choose SemEval2017 and Inspec as they contain documents of short length resembling the learning content in e-learning platforms. We also show the performance of our method on a dataset containing long documents such as SemEval2010. The statistics of the datasets are shown in Table 4.2. Since our algorithm is completely unsupervised, we evaluate on all the documents in each of these datasets.

4.3.2 Baselines and Variants of the Proposed Method

In this section, we describe the variants of the proposed CoTagRank algorithm and other baselines. In the proposed CoTagRank algorithm, a complete graph is formed from the phrases. The phrases and the document are represented by combining the contextualized embeddings from Universal Sentence Encoder (USE) (512-dimensional) [66] and topical vectors from Latent Dirichlet Allocation (LDA)[4]. The number of topics K was set at 500 when running LDA.

We compare $CoTagRank$ with several variants such as:

- $CoTagRankWindow$: This algorithm is a variant of $CoTagRank$ where only the phrases that co-occur in the text within a window of the specified size are connected in the graph. While $CoTagRank$ forms a complete graph of phrases, CoTagRankWindow provides a tunable parameter, the window size w, which determines the edges formed between phrases. The vector representation and the ranking method is the same as explained earlier.

- CoTagRanks2v: This algorithm is similar to $CoTagRank$ with respect to complete graph formation and ranking using Equation 4.5. However, in *CoTagRanks2v* the static sentence representation method like Sent2Vec [74] is used to project the phrases and the document to a continuous vector space.

- $CoTagRankSentenceUSE$: A variant of the $CoTagRank$ where the document and phrase are encoded using only Universal Sentence Encoder yielding 512-dimensional representations.

[4]We leveraged the sklearn implementation for LDA https://scikit-learn.org/

We also consider two variants of EmbedRank such as EmbedRankSentenceBERT and EmbedRankSentenceUSE where *bert-base-nli-stsb-mean-tokens* from sentence-transformers[5] and Universal Sentence Encoder are used respectively as vector representation methods. We compare the performance of the proposed algorithms with strong baselines such as EmbedRank (Sent2Vec)[6], SingleRank and other unsupervised AKE methods[7].

4.3.3 Metrics

We compare the proposed approach to existing state-of-the-art unsupervised keyphrase extraction algorithms, including the latest embedding based ranking approach, EmbedRank. We focus on unsupervised extraction algorithms to obviate need for manually annotated documents and faster inference. We report R@k, P@k and F1@k as metrics following existing works [72, 35] for a fair comparison. We extract top-k related keyphrases, rather than just top-1 so that related documents can be linked to each other by computing the overlap of keyphrases identified. Another goal is to index documents with related keywords to enhance search.

4.4 Results and Analysis

4.4.1 Performance

The performance comparison of the algorithms are as shown in Table 4.3. The measures used to evaluate the algorithms are Precision, Recall and F1-score. The metrics were computed using trec-eval[8]. Since the original implementation of EmbedRank did not provide an evaluation script, we use trec-eval to compute the metrics for EmbedRank. We observe different results on the Inspec dataset from those reported in the original EmbedRank paper as they evaluate only on a subset of 500 documents than all the 2000 documents in the

[5]https://huggingface.co/sentence-transformers/bert-base-nli-stsb-mean-tokens
[6]https://github.com/swisscom/ai-research-keyphrase-extraction
[7]https://bit.ly/369Ycg7
[8]https://github.com/usnistgov/trec_eval

Table 4.3: Performance comparison. † indicates significance at 0.01 level (t-test). ‡ indicates that effect size > 0.2.

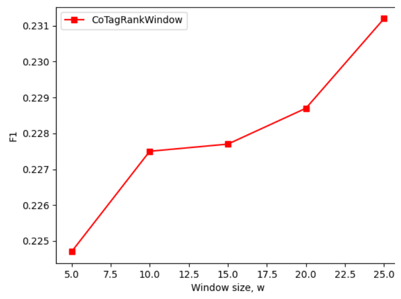
Dataset	Method	P@10	R@10	F1@10
SemEval2017	TopicalPageRank	0.3523	0.2098	0.2543
	MultiPartiteRank	0.2972	0.1758	0.2133
	SingleRank	0.3428	0.2040	0.2474
	TextRank	0.1848	0.1069	0.1326
	WordAttractionRank	0.2566	0.1482	0.1815
	EmbedRank	0.3061	0.1801	0.2195
	EmbedRankSentenceBERT	0.3329	0.1982	0.2404
	EmbedRankSentenceUSE	0.3286	0.1965	0.2381
	$CoTagRank$ (our algorithm)	**0.3911**†‡	**0.2324**†‡	**0.2819**†‡
	$CoTagRankSentenceUSE$ (our algorithm)	0.3860	0.2290	0.2779
	CoTagRanks2v (our algorithm)	0.3379	0.1990	0.2424
	$CoTagRankWindow$ (w=10) (our algorithm)	0.3797	0.2253	0.2734
	$CoTagRankWindow_{positional}$ (our algorithm)	0.3793	0.2250	0.2731
Inspec	TopicalPageRank	0.2724	0.2056	0.2260
	MultiPartiteRank	0.2210	0.1710	0.1865
	SingleRank	0.2694	0.2044	0.2239
	TextRank	0.1408	0.1020	0.1234
	WordAttractionRank	0.1778	0.1437	0.1516
	EmbedRank	0.2732	0.2034	0.2259
	EmbedRankSentenceBERT	0.2663	0.1970	0.2188
	EmbedRankSentenceUSE	0.2748	0.2049	0.2267
	$CoTagRank$ (our algorithm)	**0.2984**†‡	**0.2213**†‡	**0.2454**†‡
	$CoTagRankSentenceUSE$ (our algorithm)	0.2881	0.2150	0.2377
	CoTagRanks2v (our algorithm)	0.2372	0.1807	0.1983
	$CoTagRankWindow$ (w=10) (our algorithm)	0.2747	0.2062	0.2275
	$CoTagRankWindow_{positional}$ (our algorithm)	0.2750	0.2062	0.2276
SemEval2010	TopicalPageRank	0.0477	0.0293	0.0359
	MultiPartiteRank	**0.1757**†‡	**0.1118**†‡	**0.1352**†‡
	SingleRank	0.0457	0.0277	0.0341
	TextRank	0.0321	0.0199	0.0243
	WordAttractionRank	0.0835	0.0531	0.0641
	EmbedRank	0.0128	0.0082	0.0099
	EmbedRankSentenceBERT	0.0230	0.0137	0.0170
	EmbedRankSentenceUSE	0.0379	0.0241	0.0292
	$CoTagRank$ (our algorithm)	0.0695	0.0434	0.0530
	$CoTagRankSentenceUSE$ (our algorithm)	0.0671	0.0418	0.0511
	CoTagRanks2v (our algorithm)	0.0267	0.0169	0.0204
	$CoTagRankWindow$ (w=10) (our algorithm)	0.1337	0.0867	0.1042
	$CoTagRankWindow_{positional}$ (our algorithm)	0.1494	0.0970	0.1165
SemEval2010 (abstract and intro)	TopicalPageRank	0.1745	0.1100	0.1336
	MultiPartiteRank	0.1646	0.1044	0.1263
	SingleRank	0.1580	0.0998	0.1211
	TextRank	0.1140	0.0719	0.0872
	WordAttractionRank	0.1481	0.0949	0.1145
	EmbedRank	0.0654	0.0407	0.0496
	EmbedRankSentenceBERT	0.0844	0.0521	0.0638
	EmbedRankSentenceUSE	0.1243	0.0760	0.0933
	$CoTagRank$ (our algorithm)	0.1811	0.1134	0.1380
	$CoTagRankSentenceUSE$ (our algorithm)	0.1786	0.1121	0.1363
	CoTagRanks2v (our algorithm)	0.0852	0.0518	0.0636
	$CoTagRankWindow$ (w=10) (our algorithm)	0.1856	0.1170	0.1419
	$CoTagRankWindow_{positional}$ (our algorithm)	**0.1909**†‡	**0.1203**†‡	**0.1459**†‡

dataset [9]. Since our approach is unsupervised, we evaluate on the entire dataset.

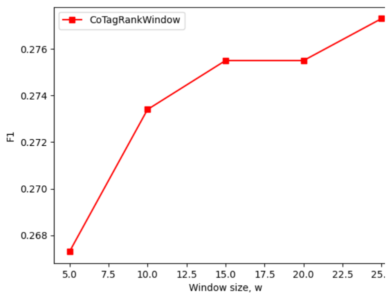
As shown in Table 4.3, the CoTagRank outperforms existing graph based and embed-

[9]Our results are close to the implementation in the project https://bit.ly/2IbbyjT which also uses trec-eval and the original EmbedRank implementation

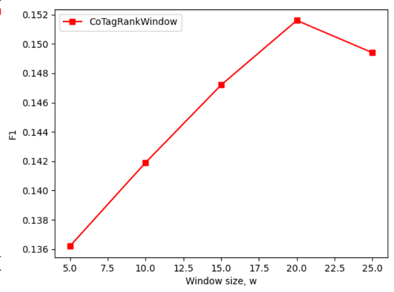
ding based unsupervised methods on two of the three datasets and on the third dataset, we get comparable results to the MultiPartiteRank algorithm. The performance gain obtained using CoTagRank over EmbedRankSentenceBERT, EmbedRankSentenceUSE and $CoTagRankSentenceUSE$ demonstrates the advantage of fusing topical information with the contextualized embeddings rather than leveraging just contextualized embeddings for phrase and document representations.



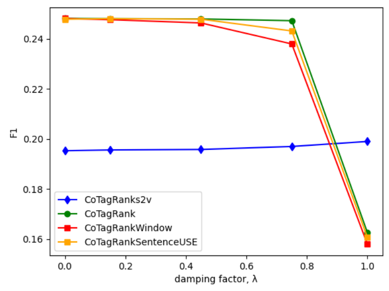
(a) F1-scores for varying w on Inspec



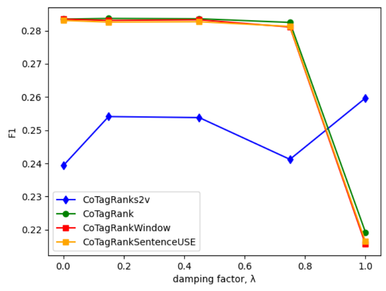
(b) F1-scores for varying w on SemEval2017



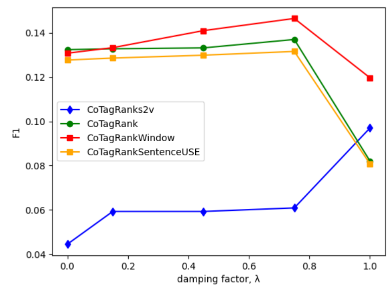
(c) F1-scores with varying w on SemEval2010 (shortened)



(d) F1-scores for different values of λ on Inspec



(e) F1-scores for different values of λ on SemEval2017



(f) F1-scores for varying λ on SemEval2010 (shortened)

Figure 4.3: Performance comparison for different hyperparameters

4.4.2 Effects of different hyperparameters

In this section, we discuss the effect of varying hyperparameters such as window size (w), damping factor (λ) and number of topics in LDA (LDA embeddings dimension) of the $CoTagRank$ algorithm and its variants. We vary the window size hyperparameter w with values 5, 10, 15, 20 and 25. The graphs in the Figure 1a and 1b show that the F1-score increases with the increase in window size for the $CoTagRankWindow$ algorithm. The window size can be set to the maximum value encompassing all phrases in the document forming a complete graph. This validates our claim that running the biased PageRank algorithm on a complete graph of phrases helps in producing high quality phrases. However, the same assumption may not hold good for longer documents, as evident from Figure 4.3c. We observe that the performance of $CoTagRankWindow$ on SemEval 2010 (abstract and

four students performed an **experiment** to calculate the **density** of a stone. while measuring the **mass** of the stone with the help of spring balance, the first student immersed the stone in water, the second student immersed it in **sulphuric acid**, the third student immersed it in kerosene and the fourth student allowed it to hang freely in air. the correct value of **mass** of the stone will be obtained by in this **experiment**, we need true value of **mass** that can be measured by suspending the stone freely in air. when we immerse the stone in some liquid, the liquid will exert an **upward buoyant force** on the stone in upward direction. this changes the reading of the spring balance and we will not get the correct value.

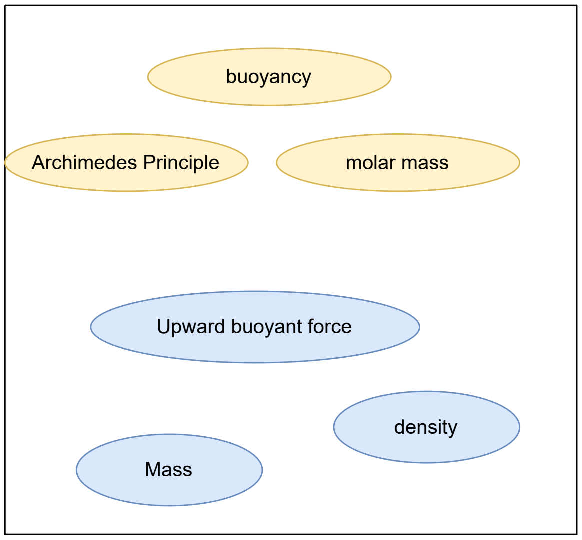


Figure 4.4: Keyphrase expansion results for an academic learning content

intro) increases with increase in window size, drops a little at $w = 25$. This indicates that forming a complete graph may not lead to the highest performance on longer documents. We also vary the damping factor λ in Equation 4.5. The values we experiment with are 0, 0.15, 0.45, 0.75, 1.0. The graphs in the Figure 4.3d and 1e show that in the proposed CoTagRank algorithm and in the variant, $CoTagRankWindow$ the performance declines with an increase in the damping factor. When damping factor is set to 1, the S_{inf} component in Equation 4.5 that contributes to *informativeness* of the phrase becomes zero, resulting in a drop in F1-score. The decrease in F1-score observed in the plots as the damping factor increases supports our claims of *informativeness* and *coherence* measures. However, we do not observe this trend in CoTagRanks2v. This may be due to the representation method used for the phrases, which do not contribute to the informativeness measure defined in this paper. From Figure 4.3f, we can observe that on SemEval2010 (abstract and intro) dataset, when λ is set to 1, there is a drop in F1-score. However, when compared to the previous two graphs, we observe that the relative drop in F1-score is low. This maybe due to the length of the documents in this dataset when compared to short length documents in Inspec and SemEval 2017.

We also vary the number of topics (LDA embeddings dimension) and observe that $CoTagRank$ and *CoTagRankWindow* achieves the highest performance when number of topics is set to 500. This is similar to the observation made in the TopicalPageRank paper [32] where the authors show that setting the number of topics to 500 gives the highest performance on the Inspec dataset.

4.4.3 Keyphrase Expansion Results

We further apply our algorithm to the task of keyphrase expansion to enrich the document with new keyphrases with the help of external knowledge sources like Wikipedia. This would help in linking related learning content in online platforms.

The keyphrases extracted from the source document using the CoTagRank algorithm serve as the seed set for the keyphrase expansion task. This is a primary advantage of *CoTagRank* as the keyphrase extraction and expansion is performed using the same algorithm unifying the fields of set extraction and expansion. Discovering latent concepts from external sources in an unsupervised manner helps in linking related documents and could also aid in better performance for document retrieval.

We use the wrapper over MediaWiki API[10] to extract relevant Wikipedia article titles for each keyphrase in the seed set. Then the expanded phrases are ranked using Equation

[10]https://pypi.org/project/wikipedia/

4.5.

Table 4.4: Performance comparison at top-10 expanded keyphrases

Dataset	**Method**	**Precision**	**Recall**	**F1**
Lecture transcripts	CoTagRank (our algorithm)	**0.5448**	**0.7270**	**0.6096**
(Khan academy)	CoTagRanks2v	0.2483	0.3424	0.2956
	$CoTagRankSentenceUSE$	0.5207	0.6950	0.5949

To demonstrate the effectiveness of this algorithm, we applied it for keyphrase expansion on 30 lecture transcripts collected from Khan academy in the science domain. The extracted phrases were given to two annotators who were undergraduate students in the Computer Science department familiar with the concepts. The task was to annotate the phrases as relevant to the document (1) or not relevant to the document (0).

The degree of agreement on relevance of keyphrases between the two annotators was measured using Cohen's kappa κ. We obtained a κ of 0.535 denoting *moderate* agreement between the annotators. A phrase is considered as a ground truth label only if both the annotators consider it to be relevant. We compute the Precision, Recall and F1 metrics as shown in Table 4.4. The F1 score of **0.6096** indicates that the proposed algorithm was able to retrieve relevant keyphrases from external knowledge sources. We observe that CoTagRanks2v and $CoTagRankSentenceUSE$ do not perform well in this task, indicating that the combination of contextualized embeddings and topic representations help in extracting better keyphrases from external knowledge sources.

Figure 4.4 shows the results of running the proposed algorithm on an academic content from Khan academy. We observe that our algorithm was able to discover interesting phrases like *Archimedes principle*, though it was not present in the source document. The new keyphrases can help in linking related learning content, where the given question in Figure 4.4 can be linked with a video explaining *Archimedes principle*. We observed that none of the other algorithms were able to retrieve *Archimedes principle*. This further reinforces the idea that apart from semantic relatedness between phrases that occur in similar contexts, their topic relatedness is also captured through our representation mechanism.

The evaluation of the proposed algorithm on this corpus demonstrates that our algorithm could also enrich the existing set of keyphrases with new keyphrases using external knowledge sources like Wikipedia. It also demonstrates the ability of the proposed algorithm to extract fine-grained concepts from academic transcripts.

4.5 Insights

A summary of insights from the results are:

- The proposed algorithm unifies the subproblems of keyphrase extraction and expansion. When applied to academic transcripts, it provides useful keyphrases for indexing the content and significant gains over existing approaches. The expansion algorithm also aids in discovering latent concepts, useful for linking related content.
- We evaluate *CoTagRank* on existing keyphrase extraction benchmarks and compare it with a wide range of existing state-of-the art unsupervised keyphrase extraction approaches. We observe that we obtain significant gains on short documents, marginal gains on medium length documents. With a slight modification to the algorithm, we observe competitive performance on large documents.

4.6 Discussions

In this paper, we proposed a novel representation and graph-based ranking algorithm, CoTagRank, for keyphrase extraction. The algorithm is currently deployed to extract academic concepts from learning content in an online learning platform. We showed that our method outperforms existing state-of-the-art unsupervised keyphrase extraction methods in shorter texts and comparable performance in longer texts. In addition, forming a complete graph of phrases outperforms window-based graph formation methods on short documents. We also demonstrated that including a simple positional bias helps further advance the performance of the algorithm on longer documents. In the future, we aim to incorporate positional embeddings and verify the performance on long texts.

We discussed the module for concept extraction and expansion which aids in making content accessible. In the next chapter, we discuss the module that aids in enriching learning content with difficulty levels. The module is pedagogically grounded as it leverages the relationship between the well-established Bloom's taxonomy and difficulty levels. We also discuss an important application of the modules discussed at the end of the next chapter.

CHAPTER 5
INTERACTIVE MULTI-TASK LABEL ATTENTION MODEL FOR QUESTION DIFFICULTY ESTIMATION

5.1 Introduction

In this chapter, we discuss a system for automated difficulty estimation of questions which is an important component for broader goals like learning objective generation and adaptive assessments [1]. We propose `QDiff`, a method for predicting the difficulty label of a question that is derived from the difficulty levels denoted as *'easy'*, *'medium'*, or *'difficult'*. We collect a set of academic questions in the Science domain from a leading e-learning platform. Bloom's taxonomy provides a mechanism for describing the learning outcomes. The different levels in Bloom's taxonomy, as observed in our dataset, are *'remembering'*, *'understanding'*, *'applying'*, and *'analyzing'*, which form the Bloom's labels. The questions are tagged with an appropriate level in Bloom's taxonomy [123] and a difficulty level. From the collected QC-Science data, we observe that the difficulty level is related to the levels in Bloom's taxonomy as shown in Table 5.1. For instance, in Table 5.1 most of the questions tagged with the *'remembering'* level of Bloom's taxonomy are categorized as being *'easy'* questions. To verify the strength of association between Bloom's taxonomy and the difficulty levels, we use the ***Cramer's V*** test since it is best suited for a large sample size. We compute the value V using the formula, $V = \sqrt{\dfrac{\chi^2}{n(min(r-1, c-1))}}$ where χ^2 is the chi-squared statistic, n is the total sample size, r is the number of rows and c is the number of columns. We obtain a value of **0.51** for V, indicating that there is a strong association between Bloom's taxonomy and the difficulty labels. Therefore, Bloom's taxonomy

[1]This chapter is partly a reproduction of a paper published at the International Conference on Artificial Intelligence in Education (AIED) [122] and concludes the set of pipelines built for content curation. It is also the final module required for the learning objective generation tool

Table 5.1: Distribution of samples across Bloom's taxonomy levels and difficulty levels (contingency table)

		Bloom's Taxonomy			
		Analyzing	Applying	Remembering	Understanding
Difficulty	Easy	756	1488	7146	4505
	Difficult	2089	2529	2518	7010
	Medium	585	980	1712	2242

labels can serve as a strong indicator for the difficulty labels and could help in the question difficulty prediction task.

As mentioned in the previous section, we observe a strong association between Bloom's taxonomy labels and difficulty labels. Hence, we propose an interactive attention model to predict the difficulty level and Bloom's taxonomy level jointly for the questions collected for classes VI to XII in the K12[2] education system.

The Bloom's taxonomy level prediction is considered as an auxiliary task, and the attention weights are computed between the vector representations of the predicted Bloom's taxonomy labels and the input vector representation through the interactive attention mechanism. We use the *hard parameter sharing* approach [124] where the backbone is a Transformer-based [125] model (like BERT [57]) with task-specific output layers on top of the backbone network. The conventional multi-task learning methods do not explicitly model the interactions between the task labels and the input. Hence we propose the interactive attention mechanism to explicitly model the interaction between Bloom's taxonomy label and the input, which enables to model the relationship between the tasks better. We observe that `QDiff` outperforms the existing baselines as measured by the macro-averaged and weighted-average F1-scores.

Following are the core technical contributions of our work:

– We propose a multi-task learning and interactive attention-based approach, `QDiff` for

[2]Kinder-garden to grade-12

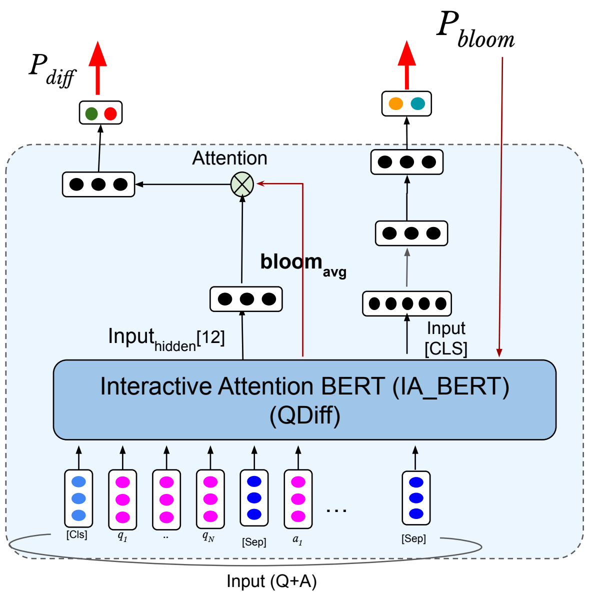


Figure 5.1: `QDiff` network architecture.

difficulty prediction, where Bloom's taxonomy predictions are used to determine the input representations using an attention mechanism.

– We evaluate the proposed method on the QC-Science dataset. We also evaluate the proposed method on another dataset QA-data [126], which consists of only difficulty labels. We show that our method trained on QC-Science dataset can be used to soft-label the QA-data dataset with Bloom's taxonomy levels. The experiments demonstrate that our method can be extrapolated to new datasets with only difficulty labels.

5.2 Methodology: An Interactive attention model for inter-task label fusion

In this section, we describe the proposed approach `QDiff` for the question difficulty prediction as the primary task and Bloom's taxonomy prediction as the secondary task. The input to `QDiff` is a corpus of questions, $C = \{q_1, q_2, ..., q_n\}$ where each q_i corresponds to a question along with a difficulty label and Bloom's taxonomy label (skill levels). Since most questions are short texts, we augment the question with the answer as auxiliary information to obtain more semantic information. Hence, we refer to the augmented question as a *'question-answer'* pair in the remainder of the paper. We show that the performance of various methods improves when using the question along with the answer rather than the question text alone.

We obtain contextualized representations for the inputs using BERT [57] followed by task-specific layers, H^{diff} and H^{bloom} where each tasks specific layer comprises of two linear layers with a non-linearity in between the two layers.

5.2.1 Contextualized Input Representations - BERT

The academic questions also have **polysemy** terms that refer to different semantics depending on the context of their occurrence in the input sentence. To tackle the mentioned problem, we use BERT, a transformer-based masked language model, for projecting the input text to the embedding space. The BERT is a language model which produces contextualized representations using a mechanism called 'Self-attention'. As mentioned in the Introduction, we classify the question difficulty levels into three classes - '*easy*', '*medium*', and '*difficult*', and Bloom's taxonomy into four classes - '*analyzing*', '*applying*', '*remembering*', and '*understanding*'.

5.2.2 Auxiliary task guided Interactive attention model

Based on the strength of association between Bloom's labels and the difficulty labels verified through Cramer's V test (V = **0.51**), we hypothesize that leveraging Bloom's taxonomy representations to compute input representations using an attention mechanism would lead to better performance in difficulty prediction. The proposed approach would help capture the relationship between the words in the input question and Bloom's taxonomy level, leading to better representations for the task of difficulty prediction as Bloom's taxonomy and difficulty levels are related. Our method also jointly learns to predict both Bloom's taxon-

Algorithm 2 Difficulty prediction

```
Input: Training set $T \leftarrow$ docs $\{q_1, ..q_n\}$, difficulty levels (labels) $y^{diff}$ and bloom's taxonomy labels
    $y^{bloom}$, test set $S$
Output: Difficulty levels for the test set $DT$
 1: Get input text embeddings , $T_{emb}, T_{pooled} \leftarrow BERT(T)$, where $T_{emb}$ represents the set of word
    embeddings
 2: Obtain Bloom's taxonomy level predictions,
    $P_{bloom} \leftarrow text\_decode(H^{bloom}(T_{pooled}))$, where $P_{bloom}$ is the text representation of predicted
    Bloom's label.
 3: Obtain the embeddings of the predicted Bloom's taxonomy,
    $bloom\_emb \leftarrow BERT(P_{bloom})$
 4: Obtain average pooled representation of $P_{bloom}$,
    $bloom\_avg \leftarrow \sum_{i=1}^{n} \frac{bloom\_emb^i}{n}$ where $n$ is the number of subwords in $P_{bloom}$.
 5: Compute attention weights,
    $\alpha_i \leftarrow softmax(f_{attn}(T^i_{emb}, bloom\_avg))$
 6: Obtain final text representations, $T_r \leftarrow \sum_{i=1}^{n} \alpha_i T^i_{emb}$
 7: Obtain difficulty level predictions, $P_{diff} \leftarrow H^{diff}(T_r)$
 8: $\mathcal{L}_{diff} \leftarrow Cross\_entropy(P_{diff}, y^{diff})$
 9: $\mathcal{L}_{bloom} \leftarrow Cross\_entropy(P_{bloom}, y^{bloom})$
10: $\mathcal{L} \leftarrow \mathcal{L}_{bloom} + \mathcal{L}_{diff}$
11: Fine-tune BERT layers and train the task specific layers to minimize $\mathcal{L}$
```

omy and the difficulty level, obviating the need for providing Bloom's taxonomy labels at inference time. Figure 5.1 shows the architecture of the proposed approach `QDiff`.

During training, as shown in Algorithm 1, the question-answer pair is first passed through a transformer-based language model BERT, to obtain contextualized word representations (T_{emb}) and the pooled representation T_{pooled} (step 1). Then the representations are passed to the task-specific output layer H^{bloom} to obtain Bloom's taxonomy level predictions (step 2). The vector representations for Bloom's taxonomy prediction are obtained using the **same BERT** model (step 3). Then the representations of subwords in P_{bloom} are averaged to obtain a fixed 768-dimensional vector representation (step 4). With the input vector representations T_{emb}, the attention mechanism generates the attention vector α_i

using Bloom's taxonomy representations $bloom\_avg$ (step 5) by

$$\alpha_i = \frac{exp(f_{attn}(T^i_{emb}, bloom\_avg))}{\sum_i^N exp(f_{attn}(T^i_{emb}, bloom\_avg))} \tag{5.1}$$

$$f_{Attn} = \tanh(T^i_{emb}.W_a.bloom\_avg^T + b_a) \tag{5.2}$$

where, tanh is a non-linear activation, W_a and b_a are the weight matrix and bias, respectively.

Then the final input representations are obtained using the attention weights α_i (step 6). The difficulty predictions are then obtained by passing the final input representation T_r through the task-specific layer H^{diff} (step 7). The BERT model acts as the backbone as its parameters are shared between the tasks. The loss function is a combination of the loss for difficulty prediction $\mathcal{L}_{diff}$ and the loss for Bloom's taxonomy level prediction $\mathcal{L}_{bloom}$ (steps 8, 9 and 10). The BERT layers are fine-tuned, and the task-specific layers are trained to minimize the combined loss (step 11).

During the *inference* phase, the contextualized representations are obtained for the test set question-answer pairs as described above. Subsequently, the softmax probability distributions of the difficulty labels are obtained by passing the contextualized representations through the H^{diff} layer. Since the BERT layers are shared between the tasks during training, the contextualized representations obtained at test time improve the performance on the task of difficulty prediction. Since we use an interaction layer with attention where the interaction between the input representations and Bloom's taxonomy label representations are captured, we also call our method as **IA_BERT** (Interactive Attention BERT).

5.3 Experiments

In this section, we discuss the experimental setup and the datasets used.

Table 5.2: Distribution of common Bloom verbs across difficulty levels

		Difficulty		
		Easy	Difficult	Medium
Bloom's Verb	Define	235	140	58
	Compare	182	223	73
	Find	744	544	253
	Choose	83	126	31
	Select	241	97	46
	Demonstrate	16	44	15
	Explain	136	144	35

5.3.1 Dataset

QC-Science: We compile the QC-Science dataset from our partner company ExtraMarks. It contains 45766 question-answer pairs belonging to the science domain tagged with Bloom's taxonomy levels and difficulty levels. We split the dataset into 39129 samples for training, 2060 samples for validation, and 4577 samples for testing. Some samples are shown in Table 4.2. The average number of words per question is 37.14, and per answer, it is 32.01.

QA-data: We demonstrate the performance of the proposed method on another dataset [126]. The dataset is labeled only with difficulty labels. We soft-label the dataset with Bloom's taxonomy levels using the Bloom's taxonomy prediction model trained on the **QC-Science** dataset. We demonstrate that the proposed model, when fine-tuned on a large enough dataset like QC-Science with labels for both tasks, can be extrapolated to datasets without Bloom's taxonomy labels. The dataset consists of 2768, 308 and 342 train, validation and test samples, respectively. The average number of words per question is 8.71, and per answer, it is 3.96.

5.3.2 Data Analysis

In this section, we briefly provide an analysis of how the training samples are distributed across Bloom's taxonomy levels and the difficulty levels. From Table 5.1, we observe that most of the samples belonging to '*remembering*' level belong to the '*easy*' category. Since some of the questions in '*remembering*' level may be tough depending on the topic, around 2518 '*remembering*' level questions are categorized as '*difficult*'. We also observe that most of the questions belonging to the '*applying*' and '*analyzing*' levels of the taxonomy are categorized to the '*difficult*' level. This supports our hypothesis that Bloom's taxonomy is a strong indicator of the difficulty level of the questions. However, we observe that the questions belonging to '*understanding*' level of Bloom's taxonomy have more samples in the '*difficult*' category than '*easy*' category. Though we observe this discrepancy, it is evident from Table 5.1 that the difference in samples between '*easy*' and '*difficult*' categories for the questions in '*understanding*' level are not as significant when compared to the distribution of samples in other Bloom's taxonomy levels.

Table 5.2 shows a distribution of common Bloom verbs across the difficulty levels. The verbs '*define*', '*find*', and '*select*' indicate the '*remembering*' level of Bloom's taxonomy. We observe that these verbs commonly occur in questions that belong to the '*easy*' difficulty level. The verbs '*demonstrate*, '*choose*', '*compare*' and '*explain*' are indicators of the '*understanding*' level of the taxonomy. We also note that these verbs commonly occur in questions that belong to the '*difficult*' category.

5.3.3 Baselines and Ablations

In this section, we briefly describe other methods used in performance comparison for the difficulty and Bloom's taxonomy prediction tasks. We first describe prior methods that solve the task of question difficulty prediction or related tasks on academic questions.

– *LDA + SVM* [38]: This method was proposed in [38] for the task of Bloom's taxonomy level prediction. Here the document-topic distributions extracted from the input documents are used as features for the SVM classifier with '*rbf*' kernel.

– *TF-IDF + SVM* [38]: In this method, the TF-IDF based features obtained from the input text are fed as input to the SVM classifier with the '*rbf*' kernel.

– *LDA + SVM & TF-IDF + SVM*[38]: We compare with this baseline proposed in [38] for the difficulty prediction task.

– *ELMo fine-tuning* [12]: Following the work of Xue *et al.*[12], we fine-tune the ELMo model for the task of difficulty prediction.

We also propose a new baseline.

– *TF-IDF + Bloom verb weights (BW)*: In this method, the samples are first grouped according to difficulty levels. Then, we extract the Bloom verbs from each sentence using the following POS tag patterns: 'VB.*', 'WP', 'WRB'. Once the Bloom verbs are extracted, we obtain Bloom verb weights as follows:

$$Bl\_W = freq(verb, label) * \frac{no.\ of\ labels}{n_l}$$

where, $freq(verb, label)$ indicates the number of times a verb appears in a label, and n_l indicates the number of labels that contain the verb. The above operation assigns higher relevance to rare verbs. Then the TF-IDF vector representation of each sentence is multiplied by the weights of the Bloom verbs contained in the sentence. Then for each difficulty level, we obtain the mean of the vector representations of the sentences (centroid). At inference time, the difficulty label whose centroid vector representation is closest to test sentence representation is obtained as output.

We also explore the following deep learning approaches.

– *Bi-LSTM with attention* and *Bi-GRU with attention and concat pooling [127].*

– *BERT cascade*: In this method, a BERT (base) model is first fine-tuned on Bloom's taxonomy level prediction followed by the task of difficulty prediction.

We also conducted several ablation studies for the proposed architecture `QDiff`.

– **Multi-Task BERT**: In this method, the interaction layer in IA_BERT is removed, and the model is trained to jointly predict Bloom's taxonomy and difficulty labels.

– **IA_BERT (B/D) (bloom/difficulty label given)**: In this method, Bloom's taxonomy label is not predicted but rather assumed to be given even at inference time. The model is trained on the objective of difficulty prediction (loss $\mathcal{L}_{diff}$) only. The Bloom's taxonomy label is considered as given when difficulty is predicted and the difficulty label is considered as given when the task is Bloom's taxonomy label prediction.

– **IA_BERT (PB) (pre-trained Bloom model)**: In this method first, a BERT model is fine-tuned for predicting Bloom's taxonomy labels alone, given the input. Then the model's weights are frozen and are used along with another BERT model, which is fine-tuned to predict the difficulty labels.

The interaction layer is the same as in IA_BERT for the last two methods mentioned above. The HuggingFace library (https://huggingface.co/) was used to train the models. All the BERT models were fine-tuned for 20 epochs with the ADAM optimizer, with learning rate (lr) of 2e-5 [57]. The LSTM and GRU based models are trained using ADAM optimizer and with lr of 0.003.

5.4 Results and Analysis

The performance comparison of various methods is shown in Table 5.3. The goal of this work is to demonstrate that the difficulty prediction and Bloom's taxonomy tasks are related, and the overall learning process can benefit from exploiting this inter-task similarity. However, question difficulty prediction is a subjective task and the automated estimation of difficulty levels are only to assist academicians. It cannot be interpreted as the final judgement of difficulty level of a question.

We use macro-average and weighted-average Precision, Recall, and F1-scores as metrics for evaluation. From Table 5.3, we can observe that most of the deep learning based

Table 5.3: Performance comparison for the difficulty prediction and Bloom's taxonomy prediction tasks. † indicates significance at **0.05** level (over Multi-task BERT). D1 - QC-Science, D2 - QA-data

		Difficulty prediction						**Bloom's level prediction**					
		Macro			**Weighted**			**Macro**			**Weighted**		
	Method	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
	LDA+SVM [38]	0.319	0.354	0.336	0.406	0.492	0.445	0.279	0.266	0.267	0.320	0.360	0.338
	TF-IDF + SVM [38]	0.471	0.415	0.440	0.510	0.532	0.520	0.421	0.341	0.377	0.413	0.410	0.411
	ELMo [12]	0.466	0.403	0.432	0.495	0.503	0.499	0.407	0.367	0.386	0.429	0.429	0.429
	TF-IDF + BW	0.426	0.437	0.431	0.486	0.429	0.456	0.330	0.346	0.338	0.364	0.344	0.342
	Simple rule baseline	0.359	0.402	0.379	0.454	0.542	0.494	0.138	0.239	0.175	0.199	0.344	0.252
D1	Bi-LSTM with attention	0.491	0.407	0.445	0.518	0.529	0.523	0.487	0.419	0.450	0.497	0.481	0.489
	Bi-GRU with attention	0.438	0.369	0.400	0.476	0.499	0.488	0.505	0.359	0.420	0.503	0.441	0.470
	BERT (base) [128]	0.499	0.450	0.473	0.530	0.550	0.539	0.484	0.459	0.471	0.494	0.502	0.498
	BERT cascade	0.494	0.454	0.473	0.530	0.550	0.539	0.470	0.441	0.455	0.486	0.486	0.486
	Multi-task BERT (ours)	0.518	0.441	0.476	0.538	0.556	0.547	0.490	0.439	0.463	0.497	0.499	0.498
	IA_BERT (`QDiff`) (ours)	**0.544**	**0.447**	**0.491**†	**0.556**	**0.564**	**0.560**†	**0.497**	0.447	**0.471**	**0.502**	**0.506**	**0.504**†
	LDA+SVM [38]	0.356	0.359	0.357	0.370	0.409	0.388	0.232	0.205	0.218	0.580	0.655	0.615
	TF-IDF + SVM [38]	0.518	0.487	0.502	0.517	0.518	0.517	0.514	0.319	0.394	0.796	0.795	0.796
	ELMo [12]	0.635	0.623	0.629	0.654	0.658	0.656	0.450	0.386	0.415	0.779	0.784	0.781
	TF-IDF + BW	0.580	0.573	0.576	0.608	0.581	0.594	0.312	0.338	0.324	0.705	0.646	0.674
	Simple rule baseline	0.236	0.341	0.279	0.233	0.392	0.292	0.131	0.200	0.158	0.429	0.655	0.518
D2	Bi-LSTM with attention	0.628	0.605	0.616	0.644	0.655	0.650	0.430	0.357	0.390	0.765	0.766	0.766
	Bi-GRU with attention	0.524	0.534	0.529	0.563	0.591	0.577	0.402	0.295	0.340	0.753	0.722	0.737
	BERT (base) [128]	0.640	0.638	0.639	0.661	0.660	0.661	0.455	0.404	0.428	0.814	0.822	0.818
	BERT cascade	0.662	0.666	0.664	0.683	0.681	0.682	0.437	0.401	0.418	0.816	0.827	0.821
	Multi-task BERT (ours)	0.664	0.644	0.654	0.681	0.687	0.684	0.389	0.365	0.377	0.799	0.804	0.802
	IA_BERT (`QDiff`) (ours)	**0.684**	**0.682**	**0.683**†	**0.702**	**0.708**	**0.705**†	**0.494**	**0.420**	**0.454**†	**0.841**	**0.830**	**0.836**†

methods outperform classical ML based methods like TF-IDF + SVM and LDA + SVM. However, we observe that TF-IDF + SVM method outperforms the ELMo baseline[12] on the QC-Science dataset. It is also evident that the transformer based methods significantly outperform the 'TF-IDF + Bloom verbs' baseline. This demonstrates that the contextualized vector representations obtained have more representational power than carefully hand-crafted features for the task of difficulty prediction.

Is the simple rule-based baseline enough?

We implement a simple rule-based baseline (Table 5.3) where the question or answer content is not considered and the difficulty label is predicted based on co-occurrence with the corresponding bloom's label alone. We form a dictionary recording the co-occurrence counts of the bloom's labels and difficulty labels in the training samples as shown in Table 5.1. For each test sample, we look up into the dictionary the entries for the corresponding bloom's taxonomy label of the test sample. Then the difficulty label with maximum co-occurrence count is chosen. We observe that this baseline performs poorly when compared to even other ML baselines from Table 5.3. This baseline performance demonstrates the need for learning based methods to analyze the given content.

Table 5.4: Ablation studies for `QDiff` (IA_BERT). D1 - QC-Science, D2 - QA-data

		Difficulty prediction						**Bloom's level prediction**					
		Macro			**Weighted**			**Macro**			**Weighted**		
	Method	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
D1	IA_BERT (`QDiff`)	0.544	0.447	0.491	0.556	0.564	0.560	0.497	0.447	0.471	0.502	0.506	0.504
	IA_BERT (B/D)	0.523	0.486	**0.503**	0.554	**0.569**	0.561	0.476	0.473	0.475	0.499	0.506	0.503
	IA_BERT (PB)	0.491	0.487	0.489	0.535	0.530	0.533	0.482	0.463	0.472	0.496	0.501	0.498
D2	IA_BERT (`QDiff`)	0.684	0.682	0.683	0.702	0.708	**0.705**	0.494	0.420	0.454	0.841	0.830	0.836
	IA_BERT (B/D)	0.682	0.688	0.685	0.702	0.696	0.699	0.458	0.429	0.443	0.837	0.842	0.839
	IA_BERT (PB)	0.642	0.641	0.641	0.667	0.652	0.659	0.449	0.424	0.436	0.825	0.825	0.825

Is the proposed approach better than baselines?

We observe that `QDiff` (IA_BERT), which jointly learns to predict Bloom's taxonomy level and the difficulty level, outperforms all the deep learning (DL) and machine learning (ML) based baselines on both datasets. It also performs better than BERT (base) [128] on the ask of difficulty prediction by advancing the weighted F1-score from 0.539 to 0.560 (+3.89%). It also outperforms Multi-task BERT by **2.37%** (weighted F1-score), which can be considered as an ablation of the proposed method without the interactive attention mechanism. This demonstrates that in addition to jointly learning to predict labels for both tasks, the interactive attention mechanism yields better representations, leading to improved performance. We also observe that the QA-data dataset `QDiff` (IA_BERT) outperforms other methods, as measured by macro and weighted F1 scores. For Bloom's label prediction, we observe that the IA_BERT (`QDiff`) leads to good results (from Table 5.3) as Bloom's labels are jointly learned and used as a signal in the attention mechanism. In addition, we also perform an experiment where we use the jointly predicted difficulty labels as signal in the interactive attention mechanism in IA_BERT for Bloom's label prediction task. We observe that the macro F1-score increases to **0.479** from 0.471 for QC-Science and also increases the weighted F1-score on QA-data to **0.852** from 0.836. This demonstrates that both tasks can benefit from each other through the interactive attention mechanism in addition to joint learning. We do not tabulate this result as our focus is difficulty prediction and mention it here for completion.

Table 5.5: Precision, Recall and F1 measures for the difficulty prediction and Bloom's taxonomy prediction tasks (using question text only).

	Difficulty prediction						**Bloom's level prediction**					
	Macro			**Weighted**			**Macro**			**Weighted**		
Method	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
LDA+SVM [38]	0.309	0.350	0.328	0.394	0.487	0.436	0.335	0.262	0.294	0.343	0.350	0.346
TF-IDF + SVM [38]	0.442	0.384	0.411	0.474	0.507	0.489	0.410	0.341	0.372	0.408	0.408	0.408
ELMo [12]	0.450	0.370	0.406	0.482	0.502	0.491	0.383	0.370	0.376	0.408	0.410	0.409
TF-IDF + Bloom verb weights	0.418	0.425	0.421	0.478	0.427	0.451	0.321	0.343	0.331	0.356	0.319	0.336
Bi-LSTM with attention	0.484	0.382	0.427	0.502	0.514	0.507	0.456	0.392	0.421	0.468	0.454	0.461
Bi-GRU with attention	0.416	0.370	0.392	0.458	0.490	0.473	0.463	0.349	0.398	0.478	0.420	0.447
BERT (base)	0.460	0.456	0.458	0.510	0.515	0.513	0.460	0.443	0.451	0.481	0.481	0.481
BERT cascade	0.468	0.441	0.454	0.508	0.529	0.518	0.438	0.423	0.430	0.456	0.464	0.460
Ensemble using MLP [129]	0.459	0.458	0.458	0.510	0.512	0.511	0.438	0.423	0.430	0.455	0.463	0.459
IA_BERT (`QDiff`) (ours)	0.468	0.442	0.455	0.510	0.531	0.520	0.448	0.375	0.409	0.457	0.466	0.461
IA_BERT (B/D) (ours)	0.473	0.458	0.465	0.516	0.529	0.522	0.474	0.448	0.461	0.489	0.496	0.493
IA_BERT (PB)	0.484	0.471	0.477	0.530	0.544	0.535	0.477	0.450	0.463	0.492	0.495	0.493
Multi-task BERT (ours)	0.471	0.458	0.464	0.519	0.530	0.524	0.461	0.421	0.440	0.473	0.469	0.470

Are the results statistically significant ?

We perform statistical significance test (*t-test*) on obtained outputs. We observe that the results obtained using `QDiff`(IA_BERT) are significant with p-value $= 0.000154$ (weighted-F1) and p-value $= 0.011893$ (macro-F1) for difficulty prediction on QC-Science dataset. We also observe that results are statistically significant on QA-data with p-value $= 0.003813$

(weighted-F1) and p-value $= 0.02248$ (macro-F1) for difficulty prediction.

Does augmenting the question with an answer lead to better performance ?

From Table 5.3, it is evident that augmenting the question with the answer provides better performance when compared to using the question text alone. When we evaluated on the QC-Science dataset using the question text alone (Table 5.5), we observed a drop in performance. For instance, `QDiff` only yielded a macro-F1 score of 0.455 on QC-Science as observed in Table 5.5 when using question text alone. The baselines also show a decline in performance, demonstrating that the answer helps provide some context for the model to perform the difficulty categorization task.

What if Bloom's taxonomy labels are randomly labeled for QA-data?

We also perform an **ablation study** on the QA-data dataset by randomly labeling the dataset with Bloom's levels instead of the proposed soft labeling method. We observe that this lowers the performance on the task of difficulty prediction. For instance, `QDiff` yields a macro F1 score of 0.656 and a weighted F1 score of 0.674 on the difficulty prediction task using the random Bloom's labels. This ablation study supports the significance of the proposed soft labeling method and the interactive attention mechanism as random labels lead to erroneous predictions of difficulty labels.

How does the ablations of IA_BERT `QDiff` perform?

We also perform several ablations studies by varying the components of the proposed method. The results are as shown in Table 5.4. We observe that using a pre-trained model for Bloom's taxonomy prediction performs poorly as it is not jointly trained on the two related tasks resulting in the error from Bloom's label prediction model propagating to the difficulty prediction task through the interactive attention mechanism. Additionally, we observe that directly feeding the Bloom's taxonomy label for the task of difficulty prediction provides gain in performance in certain scenarios as demonstrated by the second ablation in Table 5.4 for datasets D1 and D2. However, this setting is not possible in real-time scenarios as during inference, the incoming content would not be labeled with Bloom's taxonomy labels. We also observe that the performance of the original IA_BERT (`QDiff`) method is very close to the mentioned ablation study. This demonstrates that the error propagation from Bloom's label prediction is mitigated in our approach.

5.5 Insights

– The interactive attention module models the interaction between difficulty and Bloom's taxonomy prediction tasks, offering statistically significant gains in performance on the difficulty prediction task. This helps support the hypothesis of modelling inter-task similarity through attention helps in positive transfer.

– Weakly supervised labels for auxiliary task provides a useful signal in multi-task learning, as observed from results on QA-data. Although this depends on the performance of the predictor used for weak supervision. Weak predictors may result in negative transfer due to error propagation.

5.6 Discussions

In this paper, we proposed a novel method for predicting the difficulty level of the questions. The proposed method, `QDiff`, leverages an interactive attention mechanism to model the relation between bloom's taxonomy labels and the input text. We observe that `QDiff` outperforms existing methods. The results also confirm the hypothesis that modeling the interaction between the input and the task labels through an attention mechanism performs better than implicit interactions captured using only multi-task learning. Though question difficulty estimation is subjective, we observe that modeling the interaction between related tasks improves performance.

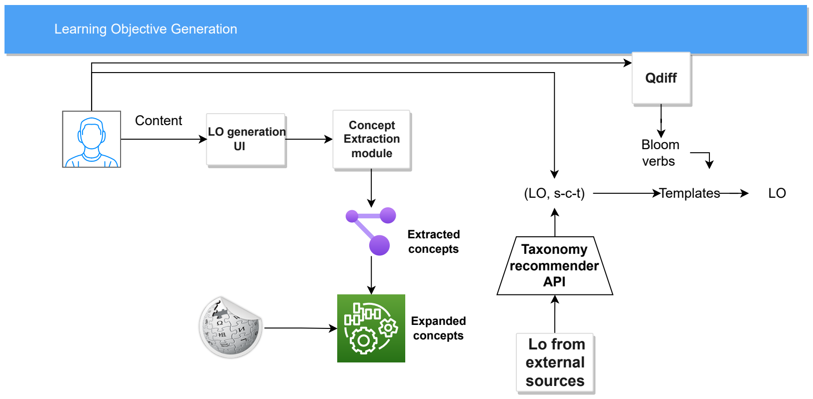
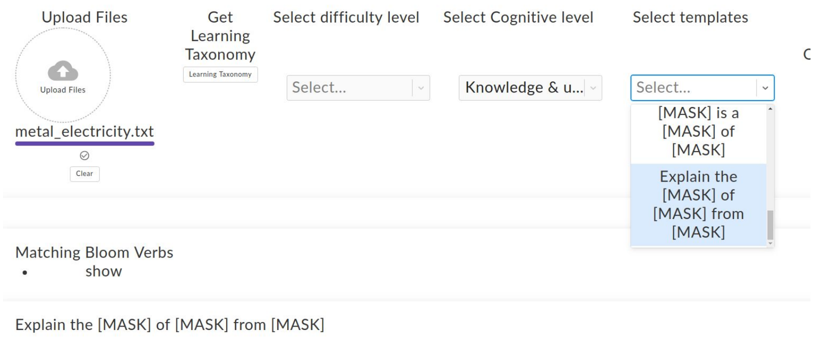


Figure 5.2: Workflow of the learning objective generation tool



Explain the [MASK] of [MASK] from [MASK]

Figure 5.3: Workflow of the learning objective generation tool

5.7 A Tool for Learning Objective Generation

We propose a tool for generating learning objectives. Learning objectives help ground the goals for the learners and also aid the academicians in understanding the effectiveness of content delivery [3]. The overview of the system workflow is shown in Figure 5.2.

Existing learning objectives (LOs) from open data sources are collected offline. They are first tagged to a common standardized taxonomy using *TagRec++*. Then, we extract and expand fine-grained concepts from the LOs and convert the corresponding positions in the LO to placeholders so as to extract templates from the learning objectives. When the

[3]https://www.rajeevelt.com/learning-objectives-importance-and-benefits-school-education/rajeev-ranjan/

user logs in to the interface and uploads the learning content, *CoTagRank* is first employed. It extracts and expands fine-grained concepts from the uploaded content. Then the content is tagged to a standardized taxonomy using *TagRec++* to map them to relevant learning objectives. Since we want to extract relevant learning objective templates for the content, we tag the content to the appropriate level in Bloom's taxonomy using *Qdiff*. The learning objective templates start with certain bloom verbs. Hence, the bloom verbs corresponding to the Bloom's level of the content can be extracted and used to retrieve relevant templates. The top-ranked templates along with concepts from the uploaded content, are presented to the user. The user can click on the concepts in order to fill the slots in the template. A screenshot of the tool is shown in Figure 5.3.

We have proposed modules to aid academicians in curating, linking and enriching content. In the next chapter, we discuss learning content enrichment through paraphrasing.

CHAPTER 6
SELF-SUPERVISED PARAPHRASING OF MATH WORD PROBLEMS

6.1 Introduction

Online assessments help gauge the understanding of the learner. In particular, for math word problems, it is pertinent that the learner is able to decipher and apply the appropriate mathematical concepts from the word problems. The concepts have to be deciphered by chunking the problem into parts and establishing the relationship between parts. Our motivation for paraphrasing math word problems is grounded in learning theory [1]. It states that learning to interpret a MWP in one's own words by changing the linguistic information is an important step in solving problems. We posit that changing the linguistic information without modifying the meaning or the solution to the problem helps improve mathematical understanding. It also helps prevent memorizing patterns to solve problems and also plagiarism by generating diverse versions of the mathematical problems.

To this end, we focus on Math word problems (MWPs) and propose the novel task of paraphrasing MWPs. We define the paraphrasing of MWPs as *changing the surface form of the original problem while preserving the underlying equation and solution.* Paraphrasing MWPs is more complex than paraphrasing general-domain text, as MWPs require preserving crucial aspects like numbers and units to ensure solution consistency. Going forward, we refer to Math word problems as "algebraic question" or "math question" alternatively. Further, word problems usually consist of multiple sentences and thus require coherence and consistency in the generated paraphrase over an extended context. Another facet crucial to our task is the diversity of the generated paraphrase due to the inherent characteristics of MWPs, wherein a change in entities along with structure yields valid paraphrases if the

[1]https://www.exinn.net/solve-it-instructional-approach-helps-students-read-for-understanding-and-paraphrase-when-solving-math-word-problems/

underlying equation is unperturbed. Although existing approaches [98, 103, 130] demonstrate competent results on general-domain corpora, they tend to hallucinate or omit key entities when applied to the task of paraphrasing MWPs, thus rendering questions unsolvable (c.f Figure 6.1). We first propose a paraphrase quality detector which would serve as a guidance for the paraphraser proposed later in this chapter. Our quality detector is named **MWPScore** which is a self-supervised method for automated scoring of paraphrases of Math Word problems [2].

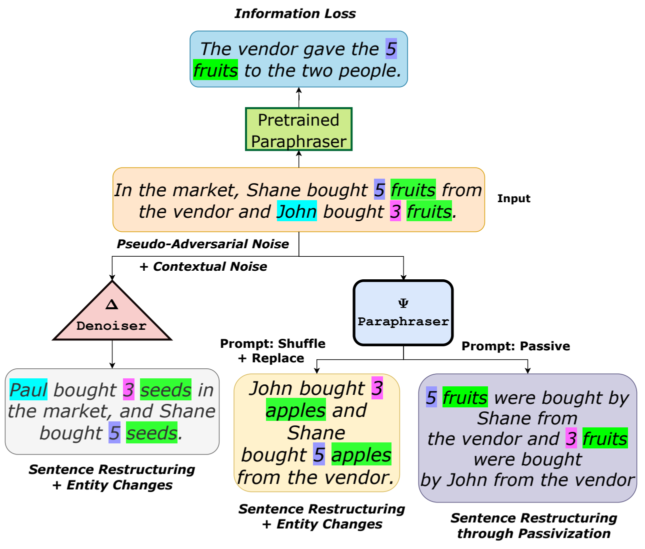

Information Loss
The vendor gave the 5 fruits to the two people.
Pretrained Paraphraser
In the market, Shane bought 5 fruits from the vendor and John bought 3 fruits.
Input
Pseudo-Adversarial Noise
+ Contextual Noise
Δ Denoiser
Ψ Paraphraser
Prompt: Shuffle + Replace
Prompt: Passive
Paul bought 3 seeds in the market, and Shane bought 5 seeds.
John bought 3 apples and Shane bought 5 apples from the vendor.
5 fruits were bought by Shane from the vendor and 3 fruits were bought by John from the vendor
Sentence Restructuring + Entity Changes
Sentence Restructuring + Entity Changes
Sentence Restructuring through Passivization

Figure 6.1: Overview of the outputs of our models (STEAD and PAP) as well as a paraphraser trained on general-domain corpora (PEGASUS fine-tuned on PAWS). The general domain paraphraser frequently loses critical information, rendering the paraphrased problem unsolvable [24]. Our models generate semantically coherent yet syntactically diverse outputs, as shown. In this instance, the denoiser generates outputs which display sentence restructuring as well as entity changes through the application of different types of noise. Similarly, given different prompts, the paraphraser generates distinct and diverse outputs, including a passivized version.

The paraphrasing task can be tackled using supervised approaches like in [132] or self-supervised approaches like in [89]. As shown in Figure 6.2, we observed that the generated paraphrases are of low quality, as critical information is lost and the solution is not preserved. Some common issues that arose for the paraphrasing models were replacement or

[2]This chapter is partly a reproduction of a paper published at ECML-PKDD [24] and a paper under review [131].

removal of numerical terms, important entities, replacement of units with irrelevant ones and other forms of information loss. These issues result in the generated question having a different solution or being rendered impossible to solve. Thus, there exists a need to automatically evaluate if a paraphrase preserves the semantics and solution of the original question. This is a *more challenging problem* than detecting similarity for general sentences. The existing state-of-the-art semantic similarity models give a relatively high score even to very low-quality paraphrases of algebraic questions (where some critical information has been lost), as seen in Figure 6.2. In Figure 6.2, our approach Paraphrase Quality Detector (ParaQD) assigns the cosine similarity as -0.999, thereby preventing the low-quality paraphrases from getting chosen. There is a need for solutions like ParaQD because poor paraphrases of algebraic questions cannot be given to the students as they are either unsolvable (as observed in the figure) or do not preserve the original solution.

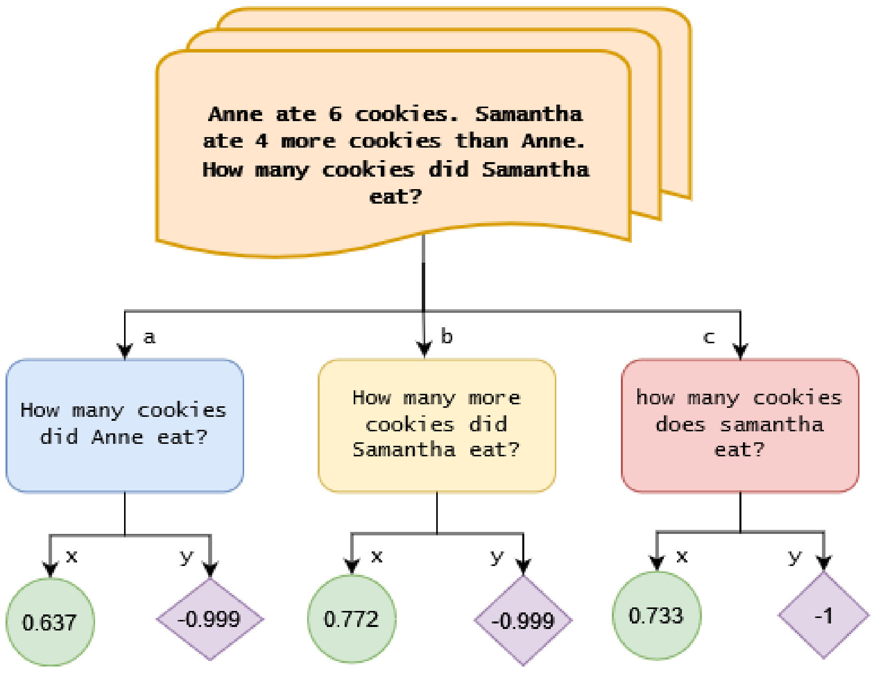

Anne ate 6 cookies. Samantha ate 4 more cookies than Anne. How many cookies did Samantha eat?
a
b
c
How many cookies did Anne eat?
How many more cookies did Samantha eat?
how many cookies does samantha eat?
x
y
0.637
-0.999
0.772
-0.999
0.733
-1

Figure 6.2: Paraphrases by SOTA generation models. *a* is output from PEGASUS fine-tuned on PAWS, *b* is from T5 fine-tuned on Quora Question Pairs dataset and *c* is from PARROT paraphraser built on T5. *x* represents the cosine similarity scores assigned by the pretrained encoder MiniLM, while *y* represents the scores with our proposed approach, ParaQD.

To tackle the issues mentioned above, we need a labelled dataset for training a proper scoring model. However, there does not exist a dataset for MWP with labelled paraphrases. Therefore, we propose multiple unsupervised data augmentations to generate positive and negative paraphrases for an input question. To model our negative augmentations, we identify crucial information in MWPs like numbers, units and key entities and design operators to perturb them. Similarly, for the positive augmentations, we design operators that promote diversity and retain the crucial information, thereby yielding a semantically equivalent MWP. On the other hand, existing augmentation methods like SSMBA [26] and UDA [88] do not capture the crucial information in MWPs. Using the positive and negative paraphrases, we train a paraphrase scoring model using triplet loss. It explicitly allows for the separation of positives and negatives to learn representations that can effectively score paraphrases. In summary, our core contributions are :

- We formulate a novel task of *detecting paraphrase quality for MWPs*, which presents a different challenge than detecting paraphrases for general sentences.
- We propose a new unsupervised data augmentation method that drives our paraphrase scoring model, *ParaQD*.
- We demonstrate that our method leads to a scoring model that surpasses the existing state-of-the-art text augmentation methods like SSMBA and UDA.
- We evaluate ParaQD using test sets prepared using operators disjoint from train augmentation operators and observe that ParaQD demonstrates good performance. We also demonstrate the zero-shot performance of ParaQD on new MWP datasets.

Code and Data are available at: https://github.com/ADS-AI/ParaQD

Our task of generating paraphrases is also unsupervised owing to lack of supervision for MWP paraphrasing. We identify two significant components of paraphrasing algebraic word problems: changing the entities (contextual changes) and changing the structure of the sentences (syntactic changes). We rigorously study and design noising functions that

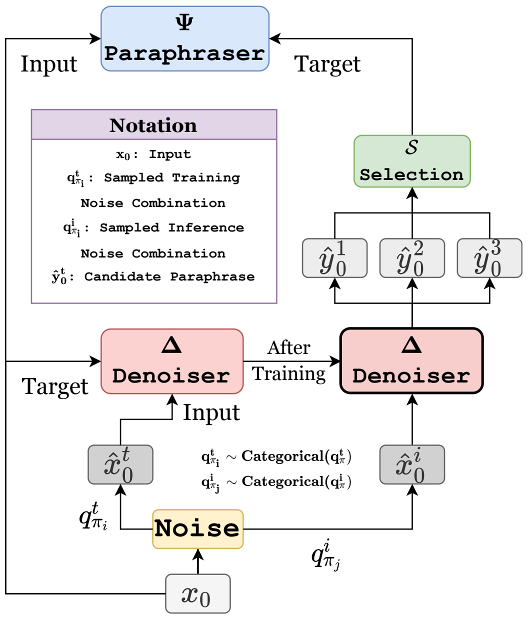


Figure 6.3: The pipeline of our proposed framework, SCANING . Given an input sample x_0, we sample a noise combination from the training bank to yield $\hat{x}^t_0$, which is given as the input for training the denoiser (STEAD) with the original sample x_0 as the reconstructive target. After training, a distinct noise combination is sampled from the inference bank, which is given to STEAD to generate multiple paraphrases $\hat{y}^1_0, \hat{y}^2_0, \ldots \hat{y}^l_0$. A selection mechanism ($QF - PCF + PAMMR$) is applied to the generated paraphrases, and the selected paraphrases are given as the target to train the paraphraser PAP , with x_0 as the input. This removes any direct dependency on the noising combination due to the existence of a direct input $\mapsto$ paraphrase mapping for PAP .

take these aspects into consideration and *exploit* them to learn a denoiser Δ that operates over both aspects to change the surface form while retaining semantic equivalence through grounded knowledge fusion. Δ is then used to generate a parallel corpus to serve as a base for learning a paraphrasing function, Ψ. Ψ is not directly dependent on the noise as it learns to map from input to paraphrase (Figure 6.3), and any supervised paraphrasing method can serve to parameterize Ψ as our method is model-agnostic. We refer to the parameterized model of Δ as STructurE Aware Denoiser (STEAD) and the parameterized model of Ψ as Prompt Aware Paraphraser (PAP). Note that these terms are used interchangeably throughout the paper.

To summarize, the core contributions of our work include:

– **Novel Task** – We propose the new task of paraphrasing algebraic word problems.

- **Noising Functions** – We define and analyze the combinations of multiple novel noising functions by modelling the search space as a composition of syntactical and contextual variations. We introduce the concept of Pseudo Adversarial Noising (PAN) to dupe the denoiser into making syntactic changes via grounding on linguistic regularities. We present a systematic overview of the induced ability from the noising functions and the reasons for their selection.

- **Novel Framework** – We design `SCANING`, a novel prompt-aware two-stage pipeline consisting of a denoiser, `STEAD` and a paraphraser, `PAP` to generate semantically rich and syntactically diverse paraphrases (even over long contexts) without the availability of reference paraphrases.

- **Evaluation** – We conduct extensive automated and human evaluations to assess the efficacy of our approach. For automated evaluation, we define new metrics (due to the new task) to quantify the soundness of the generated paraphrases. Both automated and manual results demonstrate that `SCANING` significantly outperforms all baselines.

- **Analysis** – We supply comprehensive analysis (both quantitative and qualitative) to gain an insight into the working components of the proposed framework and identify its limitations.

We release the anonymized version of our source code and data with instructions for reproducing the results at: https://anonymous.4open.science/r/SCANInG/.

6.2 Building an Automated Math Problem Paraphrase Quality Detector

In this section, we first describe the proposed method for paraphrase quality detection for math word problems. The section is divided into two components: Data Augmentation and Paraphrase Quality Detection.

6.2.1 Data Augmentation

For data augmentation, we define 10 distinct operators to generate the training set. Out of the 10, 4 are positive (i.e. information preserving) transformations, and 6 are negative (information perturbing) transformations. Our negative operators are carefully chosen after observing the common mistakes made by various paraphrasing models to *explicitly teach* the quality detection model to assign a low score for incorrect paraphrases.

Let $Q = \{Q_1, Q_2, Q_3, \ldots Q_n\}$ denote the set of questions. Each question Qi can be tokenized into sentences $Q_{i1}, Q_{i2} \ldots Q_{ip}$ where p denotes the number of sentences in question Q_i. Let an augmentation be denoted by a function f, such that $f_i(Q_j)$ represents the output of the ith augmentation on the jth question.

The function $\lambda : Q \times Q \mapsto \{0, 1\}$ represents a labelling function which returns 1 if the input (Q_i, Q_i') is a valid paraphrase, and 0 if not. Based on the design of our augmentations (explained in the next section), we work under the following assumption for the function f:

$$\lambda(Q_a, f_i(Q_a)) = \begin{cases} 1, & 1 \leq i \leq 4 \\ 0, & 5 \leq i \leq 10 \end{cases}$$

For the purposes of explanation, we will use a running example with question $\mathbf{Q_0} =$ *Alex travelled 100 km from New York at a constant speed of 20 kmph. How many hours did it take him in total?*

6.2.2 Positive Augmentations

f_1: Backtranslation

Backtranslation is the procedure of translating an example Q_i from language A to language B, and then translating it back to language A, yielding a paraphrase Q_i'. In our case, given an English question Q_i comprised of precisely p sentences $Q_{i1} \dots Q_{ip}$, we translate each sentence Q_{ij} to German Q^*_{ij}, and then translate Q^*_{ij} back to English yielding Q'_{ij} $\forall j \in \{1, 2, \dots p\}$.

$$f_1(Q_i) = concat(Q'_{i1}, Q'_{i2} \dots Q'_{ip})$$

$\mathbf{f_1(Q_0)}$: *Alex was driving 100 km from New York at a constant speed of 20 km / h. How many hours did it take in total?*

f_2: Same Sentence

Inspired by SimCSE [133], we explicitly provide the same sentence as a positive augmentation, as the standard dropout masks in the encoder act as a form of augmentation.
$\mathbf{f_2(Q_0)}$: *Alex travelled 100 km from New York at a constant speed of 20 kmph. How many hours did it take him in total?*

f_3: Num2Words

Let α be a function that converts any number to its word form. Given a question Q_i, we extract all the numbers $N_i = \{n_{i1}, n_{i2} \dots n_{ik}\}$ from Q_i. For each number $n_{ij} \in N_i$, we generate its word representation $\alpha(n_{ij})$, and replace n_{ij} by $\alpha(n_{ij})$ in Q_i to get $f_3(Q_i)$. This is done because paraphrasing models can replace numbers with their word form, and thus to ensure the scoring model does not consider it as a negative, we explicitly steer it to consider it a positive.
$\mathbf{f_3(Q_0)}$: *Alex travelled one hundred km from New York at a constant speed of twenty kmph. How many hours did it take him in total?*

f_4: UnitExpansion

Let υ be a function that converts the abbreviation of a unit into its full form. We detect all the abbreviated units $U_i = \{u_{i1}, u_{i2} \dots u_{ik}\}$ from Q_i (using a predefined vocabulary of units and regular expressions). For each unit $u_{ij} \in U_i$, we generate its expansion $\upsilon(u_{ij})$, and replace u_{ij} by $\upsilon(u_{ij})$ in Q_i. This transformation helps the model to learn the units and their expansions, and consider them as the same when scoring a paraphrase.

$\mathbf{f_4}(\mathbf{Q_0})$: *Alex travelled 100 kilometre from New York at a constant speed of 20 kilometre per hour. How many hours did it take him in total?*

6.2.3 Negative Augmentations

f_5: Most Important Phrase Deletion

The removal of unimportant words like stopwords (the, of, and) from an algebraic question will not perturb the solution or render it impossible to solve.

Thus, to generate hard negatives, we chose the most critical phrase, p_{imp} in any question, deleting which would generate Q'_i such that $\lambda(Q_i, Q'_i) = 0$. Let $\Psi : Q \mapsto P$ denote a function which returns the set of k most critical phrases $(p_1, p_2, \ldots, p_k)$ in the input Q_i.

$$p_{imp} = \operatorname*{argmin}_{p}(cossim(Q_i, Q_i \backslash p)) \qquad \forall p \in \Psi(Q_i)$$
$$f_5(Q_i) = Q_i \backslash p_{imp}$$

where $cossim$ denotes cosine similarity and $Q_i \backslash p$ denotes the deletion of p from Q_i.

$\mathbf{f_5}(\mathbf{Q_0})$: *Alex travelled 100 km from New York at a constant speed of 20 kmph. How did it take him in total?*

f_6: Last Sentence Deletion

When using existing paraphrasing models such as Pegasus, the last few words or even the complete last sentence of the input question got deleted in the generated paraphrase in some cases. Thus, to account for this behaviour, we use this transformation as a negative. More formally, let the input Q_i be tokenized into p sentences $Q_{i1}, Q_{i2} \ldots Q_{ip}$ and the sentence

Q_{i1} be tokenized into k tokens $Q_{i11}, Q_{i12} \ldots Q_{i1k}$. Then,

$$f_6(Q_i) = \begin{cases} concat(Q_{i11}, Q_{i12} \ldots Q_{i1(k-3)}) & p = 1 \\ concat(Q_{i1}, Q_{i2} \ldots Q_{i(p-1)}) & p > 1 \end{cases}$$

$\mathbf{f_6}(\mathbf{Q_0})$: *Alex travelled 100 km from New York at a constant speed of 20 kmph.*

f_7: Named Entity Replacement

Since named entities are an important part of questions, we either replace them with a random one of the same category (from a precompiled list) or with the empty string (deletion). Let $\epsilon : Q \mapsto E$ denote a function which returns a set of all named entities present in the input Q_i, such that $(e_1, e_2, \ldots, e_k) = \epsilon(Q_i)$. We randomly sample w elements $E_i = (e_a, e_b \ldots e_w)$ from $(e_1, e_2, \ldots, e_k)$ and replace/delete the entities. We set $w = rand(1, min(3, k))$ where $rand(a, b)$ represents the random selection of a number from a to b (inclusive). This restricts w from being more than 3, thus increasing the difficulty of the generated negative.

$\mathbf{f_7}(\mathbf{Q_0})$: *Sarah travelled 100 km from at a constant speed of 20 kmph. How many hours did it take him in total?*

f_8: Numerical Entity Deletion

Since numbers are critical to math questions, their removal perturbs the solution and helps generate hard negatives. Let $\nu : Q \mapsto N$ represent a function which returns a set of all numbers present in the input Q_i, such that $(n_1, n_2, \ldots, n_k) = \nu(Q_i)$. We randomly sample a subset of numbers N_i from $(n_1, n_2, \ldots, n_k)$, and sample a string s from S = *("some", "a few", "many", "a lot of", "")*. For each number $n_j \in N_i$, we replace it by s in Q_i. We set $|max(N_i)| = 2$. Similar to f_7, this makes it more challenging for the scoring model as we don't necessarily delete all the numbers, thereby generating harder negatives. This allows the model to learn that even the loss of one number renders the resultant output as an invalid paraphrase, thus getting assigned a low score.

$\mathbf{f_8}(\mathbf{Q_0})$: *Alex travelled some km from New York at a constant speed of some kmph. How many hours did it take him in total?*

f_9: Pegasus

Pegasus [134] is a transformer-based language model, fine-tuned on PAWS [135] for our purpose. Pegasus consistently gave poor results for paraphrasing algebraic questions, as shown in Figure 6.2. This provided the impetus for using it to generate hard negatives.

$\mathbf{f_9}(\mathbf{Q_0})$: = *The journey from New York to New Jersey took Alex 100 km at a constant speed.*

f_{10}: UnitReplacement

Paraphrasing models sometimes have a tendency to replace units with similar ones (such as *feet* to *inches*). Since this would change the solution to an algebraic question, we defined this transformation to replace a unit with a different one from the same category. We identified 5 categories, *C = [Currency, Length, Time, Weight, Speed]* to which most units appearing in algebraic problems belong. Our transformation was defined such that a unit u_a belonging to a particular category C_i is replaced with a unit u_b, such that $u_b \in C_i$ and $u_a \neq u_b$. For instance, *hours* could get converted to *minutes* or *days*, *grams* could get converted to *kilograms*.

Let C be the set of identified unit categories and $\Upsilon : U \mapsto U$ be a function that takes as input unit $u_a \in C_i$ and returns a different unit $u_b \in C_i$, where $C_i \in C$. Given the input Q_i containing units $U_i = (u_a, u_b \dots u_n)$, we sample a set of units $U_{is} = \{u_x, \dots u_z\}$ and replace them with $\{\Upsilon(u_i)\ \forall u_i \in U_{is}\}$ to generate $f_{10}(Q_i)$.

$\mathbf{f_{10}}(\mathbf{Q_0})$: *Alex travelled 100 m from New York at a constant speed of 20 kmph. How many hours did it take him in total?*

In the next section, we will detail our approach to training a model to detect the quality of paraphrases and how it can be used to score paraphrases.

6.2.4 Paraphrase Quality Detection

For detecting the quality of the paraphrases, we use MiniLM [136] as our base encoder (specifically, the version with 12 layers which maps the input sentences into 384-dimensional vectors)[3]. We utilize the implementation from SentenceTransformers [36], where the encoder was trained for semantic similarity tasks using over a billion training pairs and achieved high performance with a fast encoding speed[4].

We train the model using triplet loss. For each question Q_i, let the positive transformation Q_i^+ be denoted by $pos(Q_i)$ and the negative transformation Q_i^- by $neg(Q_i)$ where $pos \in (f_1, \dots f_4)$ and $neg \in (f_5, f_6 \dots f_{10})$. Let the vector representation of any question Q_i when passed through the encoder be denoted as $ENC(Q)$. Then the loss is defined as

$$Loss(Q, Q^+, Q^-) = \sum_i max(0, \alpha - dist(Q_i, Q_i^-) + dist(Q_i, Q_i^+))$$

where α is the margin parameter, $dist(Q_i, Q_i^l) = 1 - cossim(ENC(Q_i), ENC(Q_i^l))$ and $l \in \{+, -\}$. The loss ensures that the model yields vector representations such that the distance between Q_i and Q_i^+ is smaller than the distance between Q_i and Q_i^-.

At inference time, to obtain the paraphrase score of Q_i and Q_i', we use cosine similarity. Let $score : Q \times Q \mapsto [-1, 1]$ denote the scoring function, then for a pair of questions (Q_i, Q_i'):

$$\rho_i, \zeta_i = ENC(Q_i), ENC(Q_i')$$
$$score(Q_i, Q_i') = cossim(\rho_i, \zeta_i) = \frac{\rho_i \cdot \zeta_i}{|\rho_i| \cdot |\zeta_i|}$$

[1]https://bit.ly/3F2c9vH
[2]https://sbert.net/docs/pretrained_models.html

6.2.5 Experiments

All the experiments were performed using a Tesla T4 [137] and P100 [138]. All models, including the baselines, were trained for 9 epochs with a learning rate of 2e-5 using AdamW as the optimizer with seed 3407. We used a linear scheduler, with 10% of the total steps as warm-up having a weight decay of 0.01.

Datasets

The datasets used in the experiments are:

AquaRAT [139] (Apache, V2.0) is an algebraic dataset consisting of 30,000 (post-filtering) problems in the training set, 254 problems for validation and 220 problems for testing. After applying the test set operators to yield paraphrases, we get 440 samples for testing with manual labels.

EM_Math is a dataset consisting of mathematics questions for students from grades 6-10 from our partner company ExtraMarks. There are 10,000 questions in the training set and 300 in the test set. After applying the test operators, we get 600 paraphrase pairs.

SAWP (Simple Arithmetic Word Problems) is a dataset that we collected (from the internet) consisting of 200 algebraic problems. We evaluate the proposed methods in a zero-shot setting on this dataset by using the model trained on the AquaRAT dataset. After applying the test set operators, we get 400 paraphrase pairs.

PAWP (Paraphrased Algebraic Word Problems) is a dataset of 400 algebraic word problems collected by us. We requested two academicians from the partnering company (paid fair wages by the company) to manually write paraphrases (both valid and invalid) rather than using our test set operators. We use this dataset for zero-shot evaluation to demonstrate the performance of our model on human-crafted paraphrases.
Our data can also be used as a **seed set** for the task of paraphrase generation for algebraic questions.

Baselines

We compare against two SOTA data augmentation methods, UDA and SSMBA. For all the baselines, we use the same encoder (MiniLM) as for our method to maintain consistency across the experiments and enable a fair comparison.

UDA: UDA uses backtranslation and TF-IDF replacement (replacing words having a low score) to generate augmentations for any given input.

SSMBA: SSMBA is a data augmentation technique that uses corruption and reconstruction functions to generate the augmented output. The corruption is performed by masking some tokens in the input and using an encoder (such as BERT [57]) to fill the masked token.

Since the baselines are intended to generate positive paraphrases, we consider other questions in the dataset (in-batch) as negatives to train using the triplet loss. Alongside the direct implementation of UDA and SSMBA, we also compare pseudo-labelled versions of these baselines. The version of baselines without pseudo-labelling is used in all the experiments unless stated with suffix *(with pl)*.

We use a pretrained encoder (MiniLM) to first pseudo-label the samples (without being trained) and then train it using the pseudo-labelled samples. More formally, given an input Q_i and a paraphrase Q'_i, we use the encoder to determine whether Q'_i is a positive or negative paraphrase of Q_i as follows:

$$\rho_i, \zeta_i = ENC(Q_i), ENC(Q'_i)$$
$$\lambda(Q_i, Q'_i) = \begin{cases} 1 & if\ cossim(\rho_i, \zeta_i) > \iota \\ 0 & if\ cossim(\rho_i, \zeta_i) \leq \iota \end{cases}$$

where ι is the threshold for the cosine similarity, which we set to 0.8.

6.2.6 Test Set Generation

For generating the synthetic test set (for AquaRAT, EM_Math and SMWP), we define a different set of operators to generate positive and negative paraphrases to test the ability of our method to generalize to a different data distribution. For any question Q_i in the test set, we generate two paraphrases and manually annotate the question-paraphrase pairs with the help of two annotators. The annotators were instructed to mark valid paraphrases as 1 and the rest as 0. We observed Cohen's Kappa values of **0.79**, **0.84** and **0.70** on AquaRAT, EM_Math and SMWP, respectively, indicating a substantial level of agreement between the annotators.

Operator Details

We defined two positive (f_a, f_b) and three negative (f_c, f_d, f_e) test operators. For each question, we randomly chose one operator from each category for generating paraphrases. These functions are:

f_a**: Active-Passive**: We noticed that most algebraic questions are written in the active voice. We used a transformer model for converting them to passive voice[5], followed by a grammar correction model[6] on top of this to ensure grammatical correctness.

f_b**: Corrupted Sentence Reconstruction**: We corrupt an input question by shuffling, deleting and replacing tokens, similar to ROTOM [89] but with additional leniency. When corrupting the input sentence, we preserve the numbers, units and the last three tokens. This is done because if we corrupt the numbers or the units, the model cannot accurately reconstruct them and will replace them with random numbers and units. We preserve the last three tokens because corrupting them might lead the model to change the question as the last three tokens in a word problem are generally indicative of the question.

We then train a sequence transformation model (t5-base) to reconstruct the original

[5]https://bit.ly/3FbPIEu

[6]https://bit.ly/3HGOMcQ

question from the corrupted one, which yields a paraphrase.

f_c**: TF-IDF Replacement**: Instead of the usual replacement of words with low TF-IDF score [88], we replace the words with high TF-IDF scores with random words in the vocabulary. This helps us generate negative paraphrases as it removes the meaningful words in the original question rendering it unsolvable.

f_d**: Random Deletion**: Random deletion is the process of randomly removing some tokens in the input example [87] to generate a paraphrase.

f_e**: T5**: We used T5 [140] fine-tuned on Quora Question Pairs to generate negatives as it was consistently resulting in paraphrases with missing information (Figure 6.2).

Evaluation Metrics

Our main goal is to ensure the separation of valid and invalid paraphrases by a wide margin. This allows for extrapolation to unseen and unlabelled data (the distribution of scores for positive and negative paraphrases is unknown, thus threshold can be set to the standard 0.5 or a nearby value due to wider margins). It allows for the score to be used as a selection metric using maximization strategies like Simulated Annealing [103] or as reward using Reinforcement Learning [141, 142] to steer generation. To this end, along with Precision, Recall, and F1 (both macro and weighted), we compute the separation between the mean positive and mean negative scores. More formally, let the score of all (Q_i, Q_i^+) pairs be denoted by $score(Q, Q^+)$ and the score of all (Q_i, Q_i^-) pairs be denoted by $score(Q, Q^-)$ where $\lambda(Q_i, Q_i^+) = 1$ and $\lambda(Q_i, Q_i^-) = 0$. Then,

$$\mu^s \ (separation) = \mu^+ - \mu^-$$

$$\mu^l = E[score(Q, Q^l)] \ \forall \ l \in \{+, -\}$$

Test Set Details

The number of positive and negative pairs are (139, 301) in AquaRAT, (223, 377) in EM, (130, 270) in SAWP and (199, 201) in PAWP. The details of the success of test set operators are shown in the form of confusion matrices. The average precision, recall and accuracy of the operators across the datasets are 0.4, 0.59 and 0.56. The low precision is due to the inability of positive operators to generate valid paraphrases consistently, as the task of effectively paraphrasing algebraic questions is challenging. This further demonstrates the usefulness of a method like ParaQD that can be effectively used to distinguish the paraphrases as an objective to guide paraphrasing models (subsection 6.3.7).

6.2.7 Results and Analysis

Table 6.1: Precision, Recall, F1 and Separation across all methods and datasets.

Dataset	**Method**	**Macro**			**Weighted**			μ^+	μ^-	μ^s
		P	R	F1	P	R	F1			
AquaRAT	Pretrained	0.658	0.502	0.569	0.784	0.318	0.453	0.977	0.897	0.080
	UDA	0.661	0.512	0.577	0.786	0.332	0.467	0.995	0.966	0.029
	UDA (w pl)	0.659	0.507	0.573	0.785	0.325	0.460	0.996	0.973	0.023
	SSMBA	0.645	0.554	0.596	0.757	0.395	0.520	0.965	0.829	0.137
	SSMBA (w pl)	0.663	0.522	0.584	0.787	0.345	0.480	0.997	0.928	0.069
	ParaQD (ours)	0.678	0.695	**0.687**	0.762	0.625	**0.687**	0.770	-0.010	**0.780**
EM_Math	Pretrained	0.694	0.534	0.604	0.773	0.415	0.540	0.955	0.796	0.158
	UDA	0.648	0.523	0.579	0.716	0.403	0.516	0.991	0.912	0.079
	UDA (w pl)	0.683	0.587	0.631	0.751	0.485	0.589	0.963	0.751	0.213
	SSMBA	0.615	0.564	0.588	0.669	0.470	0.552	0.871	0.729	0.142
	SSMBA (w pl)	0.655	0.586	0.619	0.716	0.492	0.583	0.937	0.629	0.308
	ParaQD (ours)	0.665	0.665	**0.665**	0.708	0.622	**0.662**	0.667	0.012	**0.655**
SAWP	Pretrained	0.162	0.500	0.245	0.106	0.325	0.159	0.964	0.896	0.068
	UDA	0.557	0.514	0.535	0.636	0.358	0.458	0.958	0.912	0.046
	UDA (w pl)	0.667	0.519	0.583	0.783	0.350	0.484	0.990	0.929	0.061
	SSMBA	0.662	0.594	0.626	0.763	0.460	0.574	0.929	0.758	0.172
	SSMBA (w pl)	0.649	0.537	0.588	0.757	0.378	0.504	0.978	0.864	0.115
	ParaQD (ours)	0.636	0.645	**0.640**	0.709	0.582	**0.640**	0.656	0.068	**0.589**
PAWP	Pretrained	0.749	0.502	0.602	0.751	0.500	0.600	0.948	0.905	0.042
	UDA	0.558	0.507	0.532	0.559	0.505	0.530	0.960	0.948	0.012
	UDA (w pl)	0.668	0.510	0.578	0.669	0.507	0.577	0.988	0.961	0.026
	SSMBA	0.536	0.512	0.524	0.536	0.510	0.523	0.874	0.853	0.021
	SSMBA (w pl)	0.551	0.510	0.530	0.552	0.507	0.529	0.939	0.913	0.026
	ParaQD (ours)	0.703	0.669	**0.685**	0.703	0.668	**0.685**	0.749	0.076	**0.673**

The performance comparison and results of all methods are shown in Table 6.1. Across all datasets, for the measures macro-F1, weighted-F1 and separation, ParaQD outperforms all the baselines by a significant margin. For instance, the margin of separation in ParaQD is 5.69 times the best baseline SSMBA. To calculate the precision, recall and F1 measures, we threshold the obtained scores at the standard $\tau = 0.5$. Since this is a self-supervised method, there are no human-annotated labels available for the training and validation set. This means that the distribution of scores is unknown, and thus, the threshold can not be tuned on the validation set.

6.2.8 Performance

Our primary metric is separation as we want to separate the space of unsolvable problems more from the space of possible paraphrases of the original problem. Weighted F1 is more representative of the actual performance than macro F1 due to imbalanced data, and the results are discussed further.

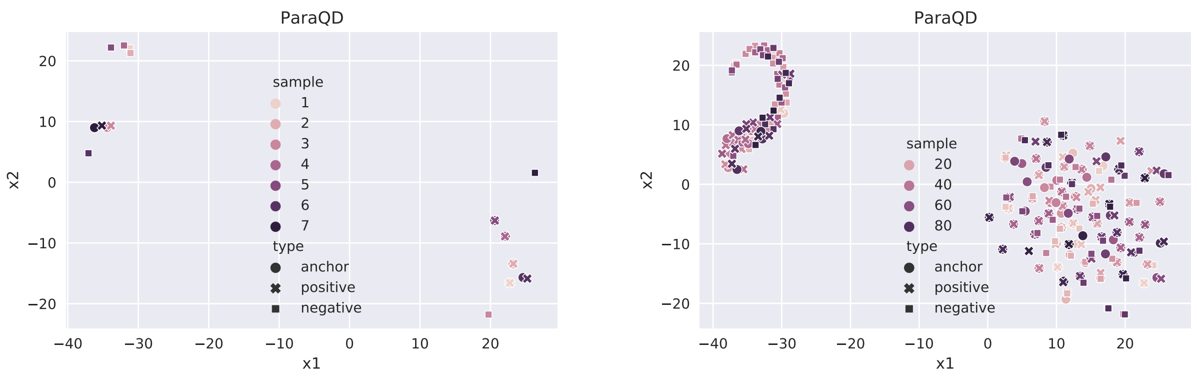
AquaRAT and EM_Math

ParaQD outperforms the best-performing baseline by 32.1% weighted F1 on AquaRAT and 12.4% weighted F1 on EM_Math. The separation achieved by ParaQD on AquaRAT is 0.78 while the best performing baseline achieves 0.137, and on EM_Math, our method achieves a separation of 0.655 while the best performing baseline achieves a separation of 0.308.

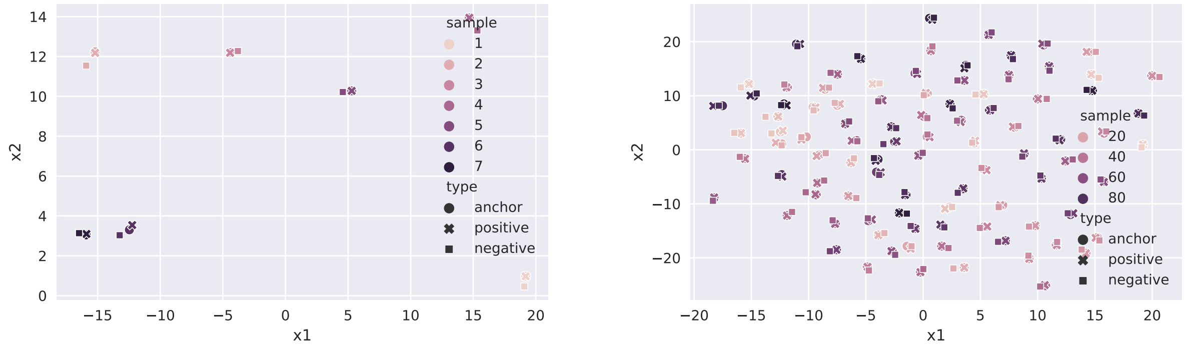
SAWP: Evaluating zero-shot performance on SAWP, ParaQD outperforms the best performing baseline by 11.5% weighted F1 and achieves a separation of 0.589 as compared to the 0.172 achieved by the best baseline. This demonstrates the ability of our method to perform well even on zero-shot settings, as the distribution of this dataset is not identical to the ones that the model was trained on.

PAWP: Our method beats the best performing baseline by 14% weighted F1 on the manually created dataset PAWP, which also consists of a zero-shot setting. It demonstrates an impressive separation of 0.673, while the best performing baseline only has a separation of 0.042. This is practically applicable as it highlights that our method can also be used to evaluate paraphrases that have been manually curated by academicians (especially on online learning platforms) instead of only on automatically generated paraphrases.

To analyze and gain a deeper insight into these results, we plotted the confusion matrices (Figure 6.6), and observed that ParaQD is able to consistently recognize invalid paraphrases to a greater extent than the baselines as it learns to *estimate the true distribution of negative samples* more effectively through our novel data augmentations.



(a) ParaQD



(b) Pretrained

Figure 6.4: Embedding plots on AquaRAT.

6.2.9 Embedding Plots

To qualitatively evaluate ParaQD, we use t-SNE to project the embeddings into a two-dimensional space as seen in Figure 6.4. We observe that the separation between anchors and negatives of triplets is minimal for the baselines, while ParaQD is able to separate them more effectively. Perhaps a more interesting insight from Figure 6.4a is that our method is able to cluster negatives together, which is not explicitly optimized by triplet loss as it does not account for inter-sample interaction. We note that our negative operators (with the

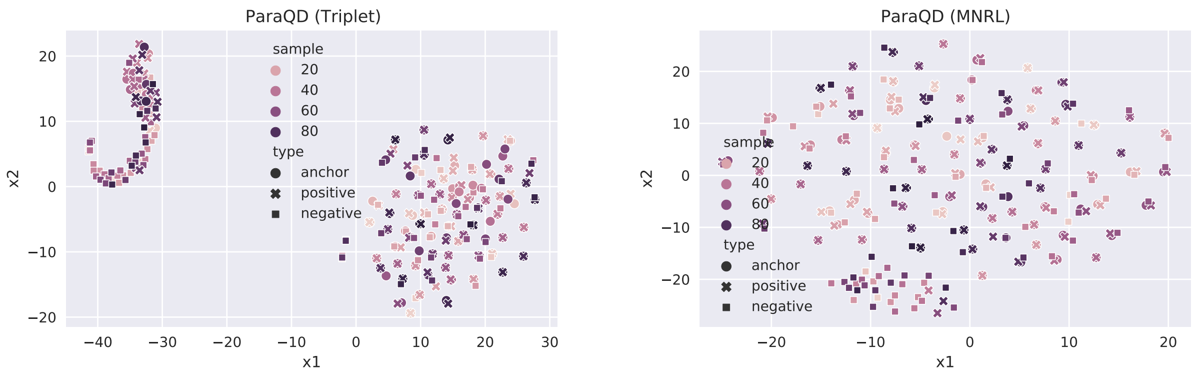
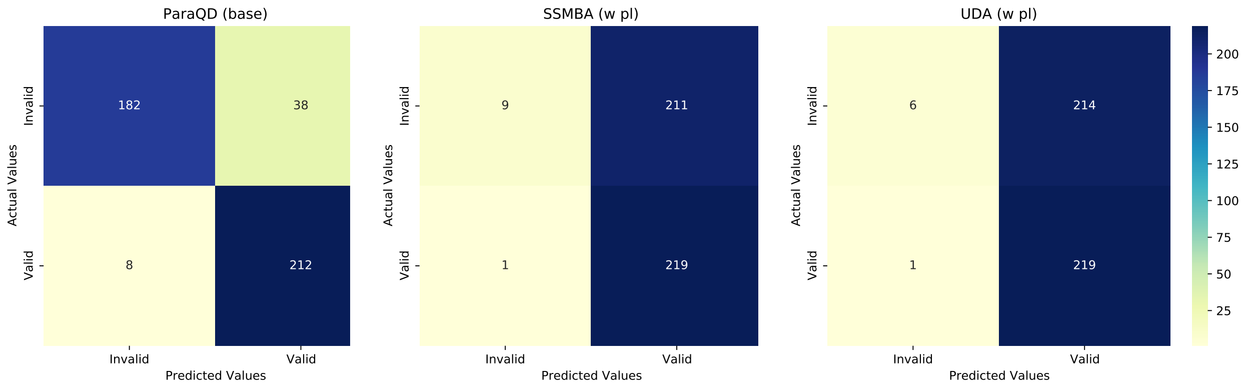
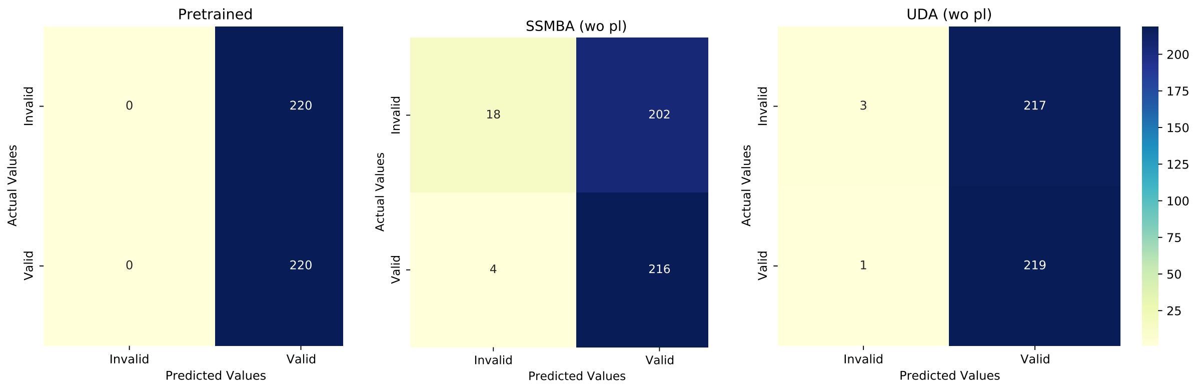


Figure 6.5: Embedding plots for different loss functions on AquaRAT



(a) ParaQD (b) SSMBA with PL (c) UDA w pl



(d) Pretrained (e) SSMBA (f) UDA

Figure 6.6: Confusion matrices for all methods on AquaRAT.

possible exception of f_7 and f_{10}) are designed to generate unsolvable problems serving as good negatives for training the scoring model (ParaQD).

Table 6.2: Summarizing the top-2 positive (Op+) and negative (Op-) operators across datasets.

Dataset	**Op+**		**Op-**	
	1	**2**	**1**	**2**
AquaRAT	f_3	f_1	f_9	f_5
EM_Math	f_4	f_1	f_9	f_8
SAWP	f_2	f_1	f_9	f_6
PAWP	f_1	f_2	f_{10}	f_9

6.2.10 Operator Ablations

To measure the impact of all operators, we trained the model after removing each operator one by one. The summary of the results is in Table 6.2. We note that f_1 (defined in Section subsubsection 6.2.2) seems to be the most consistently important operator amongst the positives, while f_9 (defined in Section subsubsection 6.2.3) is the most consistently important operator amongst the negatives. One possible reason for the success of f1 could be that it is the only positive operator that actually changes the words and sentence structure, which is replicated by our test operators and by the human-generated paraphrases.

Also, for the synthetically generated test sets (for AquaRAT, EM_Math and SAWP), since f_9 is a transformer model, it might generate paraphrases with a closer distribution (especially to f_e), but it also performs well on the human crafted paraphrases on PAWP. f_4 performs really well on EM_Math as the dataset involves more mathematical symbols, and thus the distribution of the data is such that technical operators (like f_4 and f_8) would have a more profound impact on the dataset.

The results also show that operator importance depends on the data, as certain data distributions might possess patterns that are more suitable to a certain set of operators. We also note that all operators are critical as removing any operator reduces performance for multiple datasets, thus demonstrating the usefulness of the combination of augmentations as a general framework.

Table 6.3: An ablative analysis of all methods for different seeds on AquaRAT.

Seed	Method	Macro			Weighted			μ^+	μ^-	μ^s
		P	R	F1	P	R	F1			
3407	ParaQD	0.678	0.695	**0.687**	0.762	0.625	**0.687**	0.77	-0.01	**0.78**
	UDA	0.661	0.512	0.577	0.786	0.332	0.467	0.995	0.966	0.029
	SSMBA	0.645	0.554	0.596	0.757	0.395	0.52	0.965	0.829	0.137
Seed Search	ParaQD	0.684	0.694	**0.689**	0.772	0.614	**0.684**	0.828	0.055	**0.772**
	UDA	0.659	0.503	0.571	0.784	0.32	0.455	0.998	0.985	0.013
	SSMBA	0.634	0.552	0.59	0.742	0.395	0.516	0.957	0.833	0.124

Table 6.4: Effect of the Loss Function for ParaQD (on AquaRAT)

Loss	Macro			Weighted			μ^+	μ^-	μ^s
	P	R	F1	P	R	F1			
Triplet	0.678	0.695	0.687	0.762	0.625	0.687	0.77	-0.01	**0.78**
MultipleNegativeRankingLoss	0.708	0.716	**0.712**	0.801	0.627	**0.704**	0.89	0.474	0.416

Table 6.5: An ablative analysis of all methods for 3 different encoders on AquaRAT. We observe that regardless of the encoder used, we outperform the baselines on all metrics.

Encoder	Method	Macro			Weighted			μ^+	μ^-	μ^s
		P	R	F1	P	R	F1			
all-minilm-L12-v1 (base)	ParaQD	0.678	0.695	**0.687**	0.762	0.625	**0.687**	0.77	-0.01	**0.78**
	UDA	0.661	0.512	0.577	0.786	0.332	0.467	0.995	0.966	0.029
	SSMBA	0.645	0.554	0.596	0.757	0.395	0.52	0.965	0.829	0.137
MPNet	ParaQD	0.703	0.726	**0.714**	0.785	0.659	**0.717**	0.858	0.201	**0.656**
	UDA	0.659	0.503	0.571	0.784	0.32	0.455	0.99	0.953	0.037
	SSMBA	0.66	0.508	0.574	0.785	0.327	0.462	0.985	0.94	0.045
all-minilm-L6-v2	ParaQD	0.671	0.679	**0.675**	0.758	0.598	**0.668**	0.799	0.083	**0.716**
	UDA	0.659	0.503	0.571	0.784	0.32	0.455	0.994	0.979	0.015
	SSMBA	0.661	0.513	0.578	0.786	0.334	0.469	0.992	0.941	0.051

6.2.11 Effects of Loss Functions, Encoder and Seed

We analyzed the impact of the loss function by performing an ablation with Multiple Negative Ranking Loss (MNRL) when training ParaQD. Since MNRL considers inter-sample separation, rather than explicitly distancing the generated hard negative, it is not able to provide a high margin of separation between the positives and negatives (μ^s = 0.416) as high as the triplet loss (μ^s = 0.78) but does result in a minor increase in the F1 scores. This can be observed in Figure 6.5 and Table 6.4.

We also analyzed the effects of the encoder and seed across methods on AquaRAT (Table 6.3, Table 6.5) to demonstrate the robustness of our approach. We observe that we outperform the baselines on all the metrics for three encoders we experimented with, namely MiniLM (12 layers), MiniLM (6 layers) and MPNet for different seeds.

Table 6.6: Analysis of model scores for different examples

Original	**Paraphrase**	**Label**	**ParaQD**
A bag of cat food weighs 7 pounds and 4 ounces. How much does the bag weigh in ounces?	A bag of cat food weighs 7 pounds and ounces. How much does the bag in ounces?	0	-0.922
A cart of 20 apples is distributed among 10 students. How much apple does each student get?	20 hats in a cart are equally distributed among 10 students. How much apple does each student get?	0	-0.999
A cart of 20 apples is distributed among 10 students. How much apple does each student get?	20 hats in a cart are equally distributed among 10 students. How many hats does each student get?	1	0.999
John walked 200 kilometres. How long did he walk in terms of metres?	john walked 200 centimetres. How long did he walk in terms of metres?	0	-0.999
John walked 200 kilometres. How long did he walk in terms of metres?	john walked 200 km. How long did he walk in terms of metres?	1	0.999

Error Analysis and Limitations

Does the model check for the preservation of numerical quantities?: From example 1 in Table 6.6, we observe that the number **4** is missing in the paraphrase rendering the problem unsolvable. Our model outputs a negative score, indicating it is a wrong paraphrase. This general phenomenon is observed in our reported results.

Does the model check for entity consistency?: We also observe that our model checks for entity consistency. For instance, in example 2, we observe that the paraphraser replaces *apples* with *hats* in the first sentence of the question. However, it fails to replace it in the second part of the question retaining the term *apple* which leads to a low score from ParaQD due to inconsistency. We observe from example 3 that when entity replacement is consistent throughout the question (*apple* replaced by *hats*, the model outputs a high score indicating it is a valid paraphrase.

Does the model detect changes in units?: Changing the units in algebraic word problems sometimes may render the question unsolvable or change the existing solution requiring manual intervention. For instance, from example 4 in Table 6.6, we observe that the unit *kilometres* is changed to *centimetres* in the paraphrase, which would change the equation to solve the question and by consequence the existing solution. Since we prefer solution preserving transformation of the question, ParaQD assigns a low score to this paraphrase. However, when *kilometres* is contracted to *km* in example 5, we observe that our model correctly outputs a high score.

Does the model make errors under certain scenarios?: We also analyzed the errors made by the model. We noted that samples that have valid changes in numbers are not always scored properly by the model. Thus, a limitation of this approach is that it is not robust to changes in numbers that preserve the solution. For instance, if we change the numbers 6 and 4 to 2 and 8 in Figure 6.2, the underlying equation and answer would still be preserved. But ParaQD may not output a high score for the same. We must note, however, that generating these types of paraphrases is something that is beyond the ability of general paraphrasing models. As a potential solution (in the future), we propose that numerical changes can be handled through feedback from an automatic word problem solver.

In the following section, we propose a solution for automated MWP paraphrasing which also leverages ParaQD for selecting the accurate candidate paraphrase which preserves the solution to the original problem.

6.3 Methodology: Building a Self-supervised Math Word Problem Paraphraser

In the previous section, we discussed in detail about the quality checker for paraphrasing MWPs. In this section, we discuss our self-supervised paraphrasing framework.

Due to the lack of annotated corpora for paraphrasing MWPs, we devise a self-supervised method for paraphrasing. Let the original corpus be $D = \{x^1, x^2 \dots x^n\}$ where the document x^i consisting of k tokens is represented as $(x^i_1, x^i_2 \dots x^i_k)$. We aim to generate a set of valid paraphrases $Y = \{y^1, y^2 \dots y^n\}$ without using any labelled data. We adopt a two-stage approach to accomplish this, wherein in the first stage, we learn Δ and then use it to generate a weakly-supervised parallel corpus $P = \{(x^1, \hat{y}^1), (x^2, \hat{y}^2) \dots (x^n, \hat{y}^n)\}$ for learning a paraphrasing function Ψ in the second stage (Figure 6.3). Ψ directly operates over the input-paraphrase space without carrying any direct dependency on the noise and allows us to generate a paraphrase y^i from the input x^i directly as $y^i \sim p_\Psi(y^i|x^i)$. Δ and Ψ are both parameterized by separate transformer-based neural networks [125].

6.3.1 Noising Functions

Let us denote the space of noising functions by q; instead of applying them individually to noise the input,*we sample a combination* of these noising functions from a defined categorical distribution and apply them sequentially. This is done to encourage Δ to learn to denoise multiple noises at once to increase its generalization capability to adapt to drift in the noising space. We denote the j^{th} individual noising function as q_j, the j^{th} combination as q_{π_j} and the i^{th} corrupted sample as $\hat{x}^i$, where $\hat{x}^i \sim q_{\pi_a}(\hat{x}^i|x^i)$ for some probabilistic corruption combination q_{π_a}. We refer to syntactic noise as *s-noise*, contextual noise as *c-noise* and contextual+syntactic noise as *cs-noise* for brevity and ease of understanding.

We enumerate the individual training and inference noising functions in the next two sections. We'll use the following running example e for better explainability and denote the output of the j^{th} noising function and combination as $\hat{e}^t_j \sim q^t_j(\hat{e}|e)$ and $\hat{e}^t_{\pi_j} \sim q^t_{\pi_j}(\hat{e}|e)$

respectively.

e: *Steve rode his car for 5 miles on the way home.*

6.3.2 Training Noise

As noted above, we model the noising space as a composition of both contextual and syntactic aspects. This view is taken with respect to the changes induced through the denoiser (Δ) by the noising function. For instance, the aim of an *s-noise* function is to induce syntactic variations in the output of Δ, while the aim of a *c-noise* function is to induce entity changes. We define 7 distinct individual noising functions q_i^t, and 10 combinations $q_{\pi_j}^t$ for learning Δ and analyze the intended effect of the noise on the denoiser.

s-noise

s-noise is designed to produce variations in sentence structure through word order changes. We identify the following 3 noising functions.

q_1^t : **Sentence Permutation:** Let the input x be composed of l sentences. For a sentence s, consisting of k tokens as $(s_1, s_2 \dots s_k)$, we define sentence permutation to be the selection of an index h and the rotation of s around h to yield $\hat{s} = (s_h, s_{h+1} \dots s_k, s_1 \dots s_{h-1})$. We apply this function to a stochastically selected subset of sentences in x. This function is devised to teach Δ how to permute sentences, as Δ has to revert the permuted sample to yield the original and thus learns permutation in the process.

$\hat{\mathbf{e}}_1^{\mathbf{t}}$: *Way home Steve rode his car for 5 miles on the.*

q_2^t : **Random Shuffling:** Let the input x be composed of k tokens $(x_1, x_2 \dots x_k)$. We randomly shuffle a contiguous subset of tokens $x_{sub} = (x_l, x_{l+1} \dots x_{l+p})$ of size $p + 1$, to yield $x_{shuf} = (x_{l+i}, \dots x_{l+j})$ where $0 \leq i, j \leq p$. We replace x_{sub} with x_{shuf} and repeat this procedure to generate the noised version of x.

$\hat{\mathbf{e}}_2^{\mathbf{t}}$: *Steve for rode car his 5 miles way on home the.*

q_3^t : **Complete Shuffling:** Unlike q_2^t, we completely shuffle all the tokens in the input randomly. This function teaches Δ the syntactical structure of a correct sentence as it has to learn how words chain together to form meaningful phrases to reconstruct x.

$\mathbf{\hat{e}_3^t}$: *home way miles car 5 Steve rode the his on for.*

c-noise

s-noise is designed to teach Δ to change the ordering of tokens to yield valid sentences but does not necessarily induce variations in the tokens themselves. We propose 2 *c-noise* functions that are constructed to induce contextual changes, wherein we impose additional constraints for not operating on critical entities such as numbers and units in order to preserve the solution.

q_4^t : **Templatization:** Masking [143] is a standard noising function where tokens are replaced by a masked token (usually [MASK]). We propose a novel templatization scheme, where instead of masking certain tokens with [MASK], we utilize their Part-Of-Speech (POS) tags along with a co-reference tracker as the mask. This provides a relatively stronger supervisory signal to Δ, which is necessary to maintain coherence and consistency because if multiple occurrences of one entity along with other tokens are all replaced with a singular mask, it is probable that Δ might assign different entities to the masked tokens resulting in inconsistent and invalid paraphrases.

$\mathbf{\hat{e}_4^t}$: *Steve rode **PRON1** car for 5 miles **ADP1** the **NOUN1** home.*

q_5^t : **Synonym Substitution:** We sample a subset of tokens and replace them with their synonyms. This (like q_4^t) teaches Δ to generate semantically coherent replacements for multiple tokens in the input.

$\mathbf{\hat{e}_5^t}$: *Steve **rides** his car **as** 5 miles on **part** way home.*

cs-noise

We devise 2 other noising functions, which can induce contextual and syntactic changes while applying the same constraints as *c-noise* functions.

q_6^t : **Random Deletion:** Given the input $x = (x_1, x_2 \ldots x_k)$, we stochastically select and remove a subset of tokens. This serves to teach Δ to add tokens when denoising. Removing one token induces the addition of one or multiple tokens, which can bring about both contextual and syntactical changes.

$\hat{\mathbf{e}}_6^{\mathbf{t}}$: *Steve rode his 5 miles on home.*

q_7^t : **Word Insertion:** We also add tokens to x in order to teach Δ deletion. If we only add random words, the denoiser would learn to delete tokens which do not agree with the context of the sample. To alleviate this, we also insert synonyms of the words present in the sample so that Δ can learn to delete tokens which agree with the context of the text but are unnecessary.

$\hat{\mathbf{e}}_7^{\mathbf{t}}$: *Steve **all** rode his car for **by** 5 miles on the way home.*

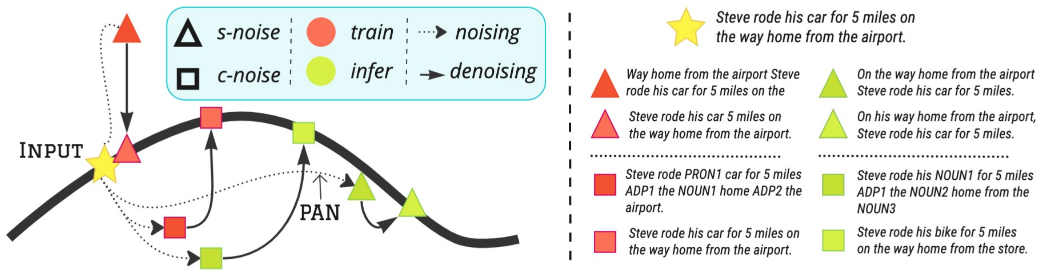

s-noise
c-noise
train
infer
noising
denoising
INPUT
PAN
Steve rode his car for 5 miles on the way home from the airport.
Way home from the airport Steve rode his car for 5 miles on the
On the way home from the airport Steve rode his car for 5 miles.
Steve rode his car 5 miles on the way home from the airport.
On his way home from the airport, Steve rode his car for 5 miles.
Steve rode PRON1 car for 5 miles ADP1 the NOUN1 home ADP2 the airport.
Steve rode his NOUN1 for 5 miles ADP1 the NOUN2 home from the NOUN3
Steve rode his car for 5 miles on the way home from the airport.
Steve rode his bike for 5 miles on the way home from the store.

Figure 6.7: *Left:* Analysis of Data Manifold for Noising Functions.

6.3.3 Inference Noise

The noising functions used for training Δ were generalized, as the aim was to learn a denoiser that could reverse multiple varieties of noise and produce coherent outputs. The goal of inference noising functions, however, is to **exploit the learned tendencies** of the denoiser and guide it to syntactically and contextually rich outputs which can serve as paraphrases on their own or be employed as weakly-supervised targets to learn Ψ. This is done via the fusion of grounded information with the noising functions to induce diversity while retaining solvability, as demonstrated in Figure 6.7. The data manifold corresponds to the *diversity distribution of valid paraphrases* for an input sample (denoted by the star).

The figure demonstrates the need for separate versions of noising functions for inference (especially for *s-noise* due to its higher degree of reversibility) as the denoised sample is rendered close to the original, displaying little diversity. Using `PAN` locates the noised sample farther from the original and *closer to the distribution of valid sentences*, ensuring that the denoiser Δ is unable to discern the type of noise applied and reverse it, which happens in the training case. *Right:* This is demonstrated with the help of an example. Here, the dark red and green triangles denote q_1^t and q_1^i, while the dark red and green squares denote q_4^t and q_3^i respectively. In the case of q_1^t, the noised sample is not grammatically sound, and the denoiser is able to implicitly identify the noise applied and reverse it (thereby losing diversity; see light red triangle). For `PAN` (q_1^i), the noised sample is close to the data manifold, so the denoiser is unable to identify the noise applied and thereby revert it, only replacing *the* with *his* and adding a *','*, which is syntactically diverse from the input (the star). Similarly, for templatization (denoted by the squares), masking certain entities yields richer outputs, which we exploit during inference (q_3^i). We must note that individual noising functions are applied here for clarity, while multiple noising functions are applied at once in practice to generate more diverse outputs.

s-noise

s-noise does not lose any information (replace/delete any token x_j^i in the input x^i) and thus has a relatively higher degree of reversibility as Δ has learned to recognize the characteristics of valid sentences and if a *s-noised* sample is not coherent, Δ reverts it back to the original state, thus losing any diversity. To combat this, we propose ***Pseudo-Adversarial Noising*** (**`PAN`**), where we *explicitly noise the input so that Δ is unable to discriminate whether s-noise has been applied and thus cannot revert it* (refer to Fig Figure 6.7 for clarity). To accomplish this, we ground the s-noise functions on linguistic regularities (like shuffling the nodes in the constituency parse tree) along with knowledge gleaned from pretrained LMs (fluency) so that the outputs display the characteristics of valid sentences.

Fluency: We model fluency as the geometric mean of the likelihood probabilities of the input sample. Specifically, we use GPT-2 [144], a pretrained auto-regressive decoder for computing the likelihood probabilities. We define the fluency of $x = (x_1, x_2 \dots x_k)$ as $fl(x) = (\prod_{j=1}^{k} p_{gpt}(x_j | x_1, x_2, \dots x_{j-1}))^{1/k}$

q_1^i : **Sentence Permutation:** Instead of permuting at any random point as in q_1^t, we discern that sentences can usually be permuted around prepositions. In the absence of a preposition within the given sentence, we permute at all possible points and select the most fluent permutation.

$\mathbf{\hat{e}_1^i}$: *On the way home Steve rode his car for 5 miles.*

q_2^i : **Phrase Shuffling:** We identify and shuffle distinct nodes in the constituency parse tree amongst each other and choose the most fluent output. $\mathbf{\hat{e}_2^i}$: *For 5 miles Steve rode his car on the way home.*

c-noise

c-noise has a relatively lower degree of reversibility as tokens get replaced with either templates or their synonyms. Thus recovery of the exact token is not always possible or desirable as similar tokens produced by the likelihood probabilities of Δ are preferred. Changes in entities (usually nouns) are more optimal than changes in pronouns, articles etc., as they lead to higher-quality paraphrases (see Figure 6.7). Thus, to bias Δ towards induction of these changes, we ground the *c-noise* functions by identifying the *object* nodes in the dependency parse tree and prioritize them for templatization/substitution.

q_3^i : **Templatization:** Object nodes are given higher priority to be templatized. The effect of this grounding on Δ is demonstrated in Figure 6.7.

$\mathbf{\hat{e}_3^i}$: *Steve rode his **NOUN1** for 5 miles **ADP1** the **NOUN2** home.*

q_4^i : **Synonym Substitution:** Similar to q_3^i, object nodes are given higher priority for substitution.

$\mathbf{\hat{e}_4^i}$: *Steve rode his **vehicles** for 5 miles on the **path** home.*

cs-noise

We retain the same operators with the same constraints as before, thus $q_5^i q_6^t$ and $q_6^i q_7^t$.

6.3.4 Noise Combinations

$q_{\pi_a}^t$: **Random Deletion + Random Shuffling + Templatization:** $\hat{\mathbf{e}}_{\pi_\mathbf{a}}^{\mathbf{t}}$: *his Steve rode for 5 miles the NOUN1 home.*

$q_{\pi_a}^i$: **Sentence Permutation + Templatization:** $\hat{\mathbf{e}}_{\pi_\mathbf{a}}^{\mathbf{i}}$: *On the way NOUN1 Steve rode his NOUN2 for 5 miles.*

6.3.5 Prompting

We utilize discrete interpretable prompts [145] for guiding the output of both Δ and Ψ. For learning Δ, we include the control for passivization by converting a fraction of the samples (20%) in the corpus to their passive form using a pretrained language model. Let the original sample be x, its passive version be x_p, the noised version be $\hat{x} \sim q^t(\hat{x}|x)$ and the discrete prompts be p_n for standard denoising and p_{pas} for passivization. Then, Δ learns to map $[p_{pas}||\hat{x}] \mapsto x_p$, and for the remaining samples, $[p_n||\hat{x}] \mapsto x$ where $[.||.]$ represents text concatenation. Practically, we set p_{pas} as *'paraphrase passive:'* and p_n as *'paraphrase:'*. p_{pas} enables Δ to *learn passivization as an auxiliary task.*

Our perspective of the noising search space also allows for prompting Ψ to add a degree of control to its outputs. Specifically, we define three distinct prompts, p_s, p_c, p_{cs} (as well as p_{pas}) for *s-noise, c-noise* and *cs-noise*. When we prompt Ψ with p_s, we expect to induce changes in the sentence structure, while when we prompt with p_c, the aim is to induce changes in entities (due to the manner in which we define the noising functions). If a combination of categories is applied, we concatenate the respective prompts. For instance, if the combination $q_{\pi_a}^i$ (Sec subsection 6.3.4) was applied, then the prompt would become $[p_s||p_c]$ This approach to prompting is a consequence of our framework design and its effects on $\Psi's$ outputs are shown in Figure 6.1.

6.3.6 STEAD

Δ is parameterized by a pretrained transformer model. Specifically, we chose BART [146] since it was pretrained with a denoising objective. Given the corpus $D^* = \{(x^1, x^1_*, p^1), \ldots (x^n, x^n_*, p^n)\}$ where $p^i \in \{p_n, p_{pas}\}$ and x^i_* is either x or x_p depending upon p^i, we train it by minimizing the negative log-likelihood loss given as

$$\mathcal{L}_\Delta = E_{(x^i, x^i_*, p^i) \sim D^*, \hat{x^i} \sim q^t_{\pi_j}(\hat{x^i}|x^i)} \\ [-\log p_\Delta(x^i_* | \hat{x^i}, p^i)]$$

6.3.7 Metrics

We apply three metrics for automated evaluation and selection of samples to learn Ψ: similarity, numeracy and diversity. Given two documents x^1 and y^1 where y^1 is the generated paraphrase of x^1, let us denote the similarity, diversity, numeracy and the PQI (overall) score between these documents as σ_{x^1,y^1}, δ_{x^1,y^1}, ν_{x^1,y^1} and λ_{x^1,y^1}. The metrics similarity and numeracy together capture the correctness and solvability of y^1 in relation to x^1, while diversity captures how contextually and syntactically different the surface forms of x^1 and y^1 are. Note that since reference paraphrases are not available to us, *we compute all the scores with respect to the original sample.*

Similarity

To measure the semantic similarity of the documents x^1 and y^1, we use a transformer-based scorer, ParaQD [24], which is designed for algebraic word problems and measures the similarity and solvability of the given documents.

Numeracy

We define numeracy between two documents x^1 and y^1 as the degree to which they share the same numerical entities. This metric proves especially useful because numbers are critical to the preservation of the solution for MWPs. More formally, let N_{d_i} denote the list of numerical entities $(N_{d1}^1, N_{d1}^2 \dots N_{d1}^k)$ in d_i. Then, we define numeracy between x^1 and y^1 as:

$$\nu_{x^1,y^1} = (\frac{|N_{x^1} \cap N_{y^1}|}{\max{(|N_{x^1}|, |N_{y^1}|)}})^p$$

where p controls the degree of penalty for hallucinating or missing a numerical entity. Note that we use a combination of σ and ν to capture correctness, as ν alone is insufficient.

Diversity

To compute the contextual and syntactic variations of x^1 and y^1, we use 1-BLEU [147] and WPD [148]. BLEU operates by leveraging the product of n-gram matches between x^1 and y^1 for various values of n to compute the surface similarity between the two documents; thus, we employ 1-BLEU to capture the diversity. This is especially useful in quantifying contextual changes in the entities but does not quantify syntactical shifts caused by permutation and shuffling. To tackle this, we use WPD, a metric proposed to capture the relative shift in the position of the words between x^1 and y^1. We use a weighted combination of these individual metrics to obtain an overall score for diversity as follows :

$$\delta_{x^1,y^1} = (1 - BLEU_{x^1,y^1}) + WPD_{x^1,y^1}$$

Paraphrase Quality Indicator (PQI)

We don't utilize the weighted arithmetic mean because if the generated paraphrase y^1 is an exact copy of x^1, then using weighted arithmetic mean would yield a high score. Thus, to compute the overall PQI score λ_{x^1,y^1}, we calculate the weighted arithmetic mean of the similarity, diversity and numeracy measures in the *log* space. This ensures that if any of σ_{x^1,y^1}, δ_{x^1,y^1} and ν_{x^1,y^1} are close to 0, then, λ_{x^1,y^1} would also be low. More formally,

$$\lambda_{x^1,y^1} = \exp(\sum_{h \in \mathcal{M}} \ln h_{x^1,y^1}); \mathcal{M} = \{\sigma, \delta, \nu\}$$

6.3.8 Data Filtering and Selection

We propose a two-stage filtering approach **QF-PCF**, where we first filter out the samples using certain thresholds on the metrics to increase the quality of the selected samples (QF), and we filter them further based on the consistency of the syntactic and contextual changes in the paraphrase with the given prompt (PCF).

QF: Quality Filtering

We define 3 thresholds $\tau_\sigma, \tau_\delta, \tau_\nu$ for σ, δ and ν respectively. We only consider the paraphrase y^1 for the next stage of filtering if it satisfies $h_{x^1,y^1} > \tau_h \forall h \in \{\sigma, \delta, \nu\}$.

PCF: Prompt Consistency Filtering

s-noise, c-noise and *cs-noise* directly correspond to the diversity metrics of WPD, (1-BLEU) and δ (overall score), respectively. Since the type of noising function has a causal relationship with the prompt, we filter out the samples where there is inconsistency between the diversity scores of the output and the prompt applied. For this, we again define 3 thresholds $\tau_s, \tau_c, \tau_{sc}$ for the prompts p_s, p_c, p_{cs}. If p_s is the given prompt, then $WPD \geq \tau_s$, if p_c is given, then $1 - BLEU \geq \tau_c$ and if p_{cs} is given, then $\delta \geq \tau_{sc}$. If a combination of the prompts is given, then the intersection of the conditions is used to filter. These thresholds are set by tuning the performance on the validation set.

Selection: Paraphrase-Adapted Maximum Marginal Relevance

For the given document q and a set of n candidates $\{p_1, p_2, \ldots p_n\}$ in bucket C, to select the top-k candidates into the bucket S ($S = \phi$ initially), inspired by MMR [149], we devise a new formulation, PAMMR, as follows:

$$
{}_{p_i \in C \setminus S}[\alpha(w_\sigma \sigma_{q,p_i} + w_\delta \delta_{q,p_i} + w_\nu \nu_{q,p_i}) \\
+ (1-\alpha) \min_{s_j \in S} \delta_{p_i,s_j}] \ni w_\sigma + w_\delta + w_\nu = 1
$$

where α is a control between the relevance and inter-sample diversity of the candidate with the already selected documents S. This formulation helps ensure that the selected documents are semantically similar and syntactically diverse with respect to the original q but are not syntactically similar to each other.

6.3.9 `PAP`

Similar to `STEAD` , Ψ is parameterized by BART, although any sequence generation model can be used. Given the weakly-supervised selected corpus $P = \{(x^1, y^1, p^1), (x^2, y^2, p^2) \ldots (x^m, y^m, p^m)\}$ where p^i denotes the prompt, we train Ψ by minimizing the negative log-likelihood loss given by:

$$
\mathcal{L}_\Psi = E_{(x^i, y^i, p^i) \sim P}[-\log p_\Psi(y^i | x^i, p^i)]
$$

6.4 Experiments

All the experiments were performed using 4 $A100$ [150] and 2 $P100$ [138] GPUs. All models, including the baselines, were trained for 15 epochs with the initial learning rate of

$8e-5$ using AdamW as the optimizer and a linear scheduler, with 10% of the total steps as warm-up having a weight decay of 0.03. Training the model took an average time of 1 hour in the distributed setup with a combined batch size of 128 (32 per GPU node). We used the base version of BART ($140M$ parameters) for initializing both `STEAD` and `PAP` . The decision to utilize a relatively smaller language model like BART ($140M$ parameters) was also influenced by deployability considerations.

6.4.1 Datasets

The datasets used in the experiments are:

• **AquaRAT** [139] (Apache, V2.0) is an algebraic dataset consisting of $30,000$ problems in the training set for the first phase and $40,000$ questions in the second phase. There are 254 problems for validation and 215 problems for testing.

• **MathEM** is a proprietary dataset from our industry partner consisting of mathematics questions for students from grades 6-10. There are 5000 and 7000 questions in the training set for the first and second phase. There are 300 questions in the validation set and 200 in the test set.

• **SAWP** [24] is a dataset consisting of 200 algebraic problems (without paraphrases). We evaluate the proposed methods in a zero-shot setting (without training) on this dataset using the models trained on AquaRAT.

• **PAWP** [24] is a dataset consisting of 400 algebraic word problems. Similar to $SAWP$, we use this dataset for zero-shot evaluation.

6.4.2 Comparative Systems

We bifurcate the comparative systems into two distinct categories: (i) augmentation-based (ii) paraphrasers. The former utilize augmentations to generate parallel data which is leveraged to train BART to ensure consistency and fairness of comparison with our method. These approaches are not specifically designed for paraphrasing. The latter systems are explicitly designed for the task of paraphrasing. We compare and contrast `SCANING` against both categories of methods to assess its efficacy.

Augmentation-based Systems

• **UDA**: We implement the unsupervised data augmentation [151] following the original settings in the paper to generate a parallel corpus. UDA leverages operations like back-translation, TF-IDF based word replacement for augmentation. We train BART to learn the mapping from input to paraphrase. • **SSMBA**: We leverage the SSMBA augmentation method [102] to generate parallel sentences for training. • **Rotom**: We use the ROTOM [89] framework to generate paraphrases and train BART using it. During inference, we go from the original sample to paraphrase, as the results degrade if we corrupt the input.

Paraphrasing systems

• **SynPG**: We utilize SynPG [152], an unsupervised encoder-decoder based model for paraphrasing, which works by disentangling the syntactic and semantic components. • **UPLM**: We reproduce the unsupervised paraphrase generation using pre-trained language models [130] with the original parameter settings. **UPSA**: In this approach [103], simulated annealing is employed to search the space for local edits to form valid paraphrases. We follow the original hyperparameter settings.

6.5 Results and Analysis

6.5.1 Comparison with other systems

We observe that `SCANING` outperforms the baselines by a significant margin measured by the PQI metric in Table 6.7. For instance, the percentage improvement in PQI over the best baseline on AquaRAT is **35%**, on MathEM is **76%** while on the zero-shot datasets it is over **35%**. We also note that our approach's standard deviation is lower than the baselines, denoting consistency (c.f. Table 6.7)

Table 6.7: Automated evaluation across all methods (mean ± std. deviation).

Dataset	Method	Similarity	Diversity	Numeracy	PQI
AquaRAT	SSMBA + BART	0.92 ± 0.27	0.08 ± 0.05	0.9 ± 0.23	0.4 ± 0.21
	Rotom + BART	0.89 ± 0.32	0.22 ± 0.06	0.71 ± 0.35	0.53 ± 0.21
	UDA + BART	0.98 ± 0.15	0.03 ± 0.03	0.97 ± 0.13	0.22 ± 0.23
	UPLM	0.72 ± 0.45	0.33 ± 0.17	0.66 ± 0.38	0.44 ± 0.29
	UPSA	0.49 ± 0.28	0.68 ± 0.06	0.03 ± 0.12	0.09 ± 0.16
	SynPG	0.55 ± 0.41	0.54 ± 0.07	0.04 ± 0.07	0.13 ± 0.19
	Denoiser (ours)	0.98 ± 0.14	0.29 ± 0.09	0.97 ± 0.13	**0.70** ± 0.11
	Paraphraser (ours)	0.98 ± 0.14	0.29 ± 0.09	1.00 ± 0.04	**0.72** ± 0.11
MathEM	SSMBA + BART	0.93 ± 0.26	0.04 ± 0.06	0.95 ± 0.21	0.23 ± 0.25
	Rotom + BART	0.95 ± 0.22	0.08 ± 0.08	0.93 ± 0.21	0.34 ± 0.27
	UDA + BART	0.96 ± 0.21	0.02 ± 0.04	0.97 ± 0.15	0.09 ± 0.17
	UPLM	0.67 ± 0.44	0.41 ± 0.20	0.64 ± 0.43	0.43 ± 0.33
	UPSA	0.33 ± 0.35	0.59 ± 0.11	0.24 ± 0.41	0.09 ± 0.19
	SynPG	0.56 ± 0.39	0.50 ± 0.11	0.21 ± 0.40	0.14 ± 0.26
	Denoiser (ours)	0.99 ± 0.08	0.36 ± 0.12	0.99 ± 0.07	**0.76** ± 0.08
	Paraphraser (ours)	0.98 ± 0.08	0.32 ± 0.11	0.99 ± 0.08	**0.73** ± 0.08
SAWP	SSMBA + BART	0.93 ± 0.25	0.06 ± 0.05	0.97 ± 0.13	0.37 ± 0.23
	Rotom + BART	0.85 ± 0.35	0.22 ± 0.06	0.75 ± 0.33	0.52 ± 0.23
	UDA + BART	0.96 ± 0.20	0.03 ± 0.04	0.99 ± 0.10	0.19 ± 0.23
	UPLM	0.79 ± 0.40	0.31 ± 0.11	0.76 ± 0.35	0.51 ± 0.28
	UPSA	0.49 ± 0.27	0.68 ± 0.06	0.04 ± 0.11	0.10 ± 0.17
	SynPG	0.43 ± 0.40	0.56 ± 0.06	0.04 ± 0.12	0.09 ± 0.17
	Denoiser (ours)	0.98 ± 0.12	0.33 ± 0.09	0.99 ± 0.09	**0.74** ± 0.11
	Paraphraser (ours)	0.99 ± 0.07	0.30 ± 0.09	1.00 ± 0.04	**0.73** ± 0.08
PAWP	SSMBA + BART	0.91 ± 0.28	0.06 ± 0.05	0.96 ± 0.16	0.34 ± 0.24
	Rotom + BART	0.88 ± 0.33	0.22 ± 0.06	0.75 ± 0.34	0.53 ± 0.22
	UDA + BART	0.96 ± 0.21	0.03 ± 0.04	0.99 ± 0.09	0.18 ± 0.23
	UPLM	0.71 ± 0.45	0.32 ± 0.13	0.70 ± 0.39	0.45 ± 0.30
	UPSA	0.51 ± 0.27	0.67 ± 0.06	0.06 ± 0.16	0.13 ± 0.19
	SynPG	0.46 ± 0.40	0.55 ± 0.06	0.04 ± 0.10	0.11 ± 0.18
	Denoiser (ours)	0.98 ± 0.13	0.33 ± 0.09	0.99 ± 0.09	**0.74** ± 0.11
	Paraphraser (ours)	0.99 ± 0.05	0.31 ± 0.08	1.00 ± 0.00	**0.74** ± 0.06

6.5.2 Performance on individual facets

We also report the performance on individual facets like similarity, diversity and numeracy elaborated in subsection 6.3.7. We observe that our method gives consistently high diversity while preserving similarity and numeracy. SynPG, Rotom, UPSA and UPLM also have a high diversity score, but manual examination of the outputs shows that they frequently omit information or begin hallucinating, thus lowering their similarity score. The hallucination effect is very pronounced for both SynPG and UPSA, where we observe that the *numeracy* measure is extremely low, rendering the problem unsolvable. Other baselines like UDA + BART have low diversity and high similarity because they primarily copy the input.

6.5.3 Human Evaluation

Table 6.8: Human evaluation results on AquaRAT for our framework and the top-2 baselines. We define Solvability, Diversity and Fluency as the metrics for evaluation. Mean ± std. deviation is reported. Higher is better for all metrics.

Method	**Solvability**	**Diversity**	**Fluency**
PAP (Ψ)	**3.59** ± 1.23	**3.96** ± 0.73	3.85 ± 0.77
STEAD (Δ)	3.52 ± 1.19	3.86 ± 0.71	**3.94** ± 0.78
Rotom	1.85 ± 1.54	1.95 ± 0.79	2.68 ± 0.88
UPLM	1.81 ± 1.66	2.27 ± 1.04	2.72 ± 1.31

We manually evaluate our methods and the top-2 baselines on AquaRAT using three metrics: solvability, diversity and fluency, as shown in Table 6.8. For the solvability metric the evaluators were requested to look for any loss of information that may render the problem unsolvable. The loss of information could include numbers, metric units or critical subject and object information. The scores are assigned in the range of 0-5 for all measures. We demonstrate that both Δ and Ψ comprehensively outperform the baselines on all metrics. We ask the evaluators[7] to rate each paraphrase on a scale of 0-5 for all the three

[7]The annotators are aged between 20 and 30 with Computer Science degrees and are able to solve MWPs efficiently.

measures of solvability, diversity and fluency in a blind manner. The solvability metric ensures that the solution of the paraphrase is the same as the source problem, while diversity oversees that the generated paraphrase differs in form from the original. Finally, fluency ensures that inter-sentence coherence is maintained and that the generated paraphrase is grammatically correct. For each sample, we average the responses received for that sample for all three measures mentioned above. The final reported value for each measure is the mean across samples, along with the standard deviation. We compute Cohen's Kappa (κ) to measure the inter-annotator agreement. We report average κ values of 0.62 for solvability, 0.56 for diversity and 0.56 for fluency, indicating *substantial* agreement for solvability and moderate agreement for the other two measures.

From Table 6.8, we observe that both the paraphraser and the denoiser give substantially more solvable and diverse outputs than the baselines, along with an increased degree of fluency. For solvability, we observe that the standard deviation is high for all methods but is considerably higher for the baselines (especially given the low mean). This denotes the ability of our proposed method to generate higher-quality paraphrases more frequently. However, the results indicate that there is still scope for improvement in the solvability aspect, as the aim is to generate diverse yet solvable outcomes consistently. We must note that out of the baselines (and other unsupervised methods), *ours is the only method that can bring changes in the sentence structure and word order consistently.* The diversity in the baselines is usually due to hallucination or the addition/deletion of some common words, while we also perform higher order restructuring through injection of `PAN` along with the use of prompting and passivization.

6.5.4 Ablation Study

We perform multiple ablations to identify the functionality of various aspects of our method empirically. We train Ψ without prompts, without PCF, using T5 as the base model (checking for model-agnosticism), and going from the paraphrases generated by the denoiser to

Table 6.9: Ablation study for both `STEAD` and `PAP` on AquaRAT. We extract out individual components in the pipeline to identify their impact and provide validation for our design choices.

Method	**S**	**D**	**N**	**PQI**
`STEAD`	0.98	0.29	0.97	**0.7**
- inference noise	0.93	0.30	0.89	0.65
`PAP`	0.98	0.29	1.00	**0.72**
- prompts	0.97	0.24	0.97	0.67
- PCF	0.99	0.28	0.99	0.71
- BART + T5	0.97	0.26	0.98	0.69
+ P2O	0.98	0.23	0.96	0.65

the original question (P2O). The results (shown in Table 6.9) validate our design choices.

We also use training noise on the test set for Δ and observe poorer results.

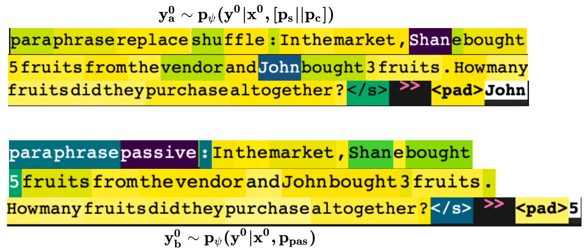

$\mathbf{y_a^0 \sim p_\psi(y^0|x^0, [p_s||p_c])}$
paraphrase replace shuffle : In the market , Shane bought
5 fruits from the vendor and John bought 3 fruits . How many
fruits did they purchase altogether ? </s> >> <pad> John
paraphrase passive : In the market , Shane bought
5 fruits from the vendor and John bought 3 fruits .
How many fruits did they purchase altogether ? </s> >> <pad> 5
$\mathbf{y_b^0 \sim p_\psi(y^0|x^0, p_{pas})}$

Figure 6.8: Feature Attribution for the same input with different prompts (*paraphrase replace shuffle:* and *paraphrase passive:*) given by Ψ where darker shade denotes higher feature salience. In case of the latter prompt, a high attribution is given to the prompt word *passive* denoting its importance as the model needs to restructure the output accordingly (passivization). Note the higher salience given to the first generated token (*John* and *5* respectively) for both prompts.

6.5.5 Prompting

We present prompt outputs in Figure 6.1 and visualize the **feature attribution** using Integrated Gradients (integral of gradients is approximated by a Riemann sum) for different prompts to increase interpretability and gain a deeper insight in Figure 6.8. In Figure 6.1, when given the prompt for syntactic and contextual noise, `PAP` replaces entities (*fruits* with *apples*) and also shuffles the structure of the input. However, when we prompt it for passivization, the model restructures the output completely and converts it into passive form.

As per Figure 6.8, note that when giving the prompt $[p_s||p_c]$, the saliency of the prompt words is relatively lower, while the named entities Shane and John are given higher weightage when generating the first token (*John*). This is because the first token (in the case without passivization) could have been either name, and the sentence structure would have followed from there. However, in the case of the passive prompt, the highest saliency is given to the prompt itself (especially the word *passive*). This is because, in this case, the prompt will dictate the structure of the output, as passivization requires high-level restructuring of the input. We also observe that the number *5* receives a much higher weightage than the previous case, as it is the first generated token and thus requires higher attribution. This example demonstrates that changing one word in the prompt while keeping the input the same can change the internal salience assigned by the model to the input tokens and thus allow it to generate the output according to the salience assigned.

6.5.6 Qualitative/Error Analysis

We present some sample outputs for both `STEAD` and `PAP` in Table 6.10. From the first 3 rows, we can observe how `STEAD` is leveraging the noise to produce diverse paraphrases. The last 4 outputs are generated by `PAP` . We can observe in the 4^{th} and 6^{th} sample that the passivization prompt is being followed, but the prompt for entity changes is not being followed, as the entities remain consistent. In the last row, we can also observe that ψ sometimes misunderstands entities (in this case, it considers A as the article *A*). This mistake leads the model to paraphrase in a way that completely destroys the solution. This type of error is not recognized and assigned a high score by all automated metrics, but during manual evaluation, the annotators were told to rate it very low (0 or 1) in solvability. This type of error is also induced in the denoiser (infrequently) by functions such as deletion and replacement due to their higher degree of irreversibility.

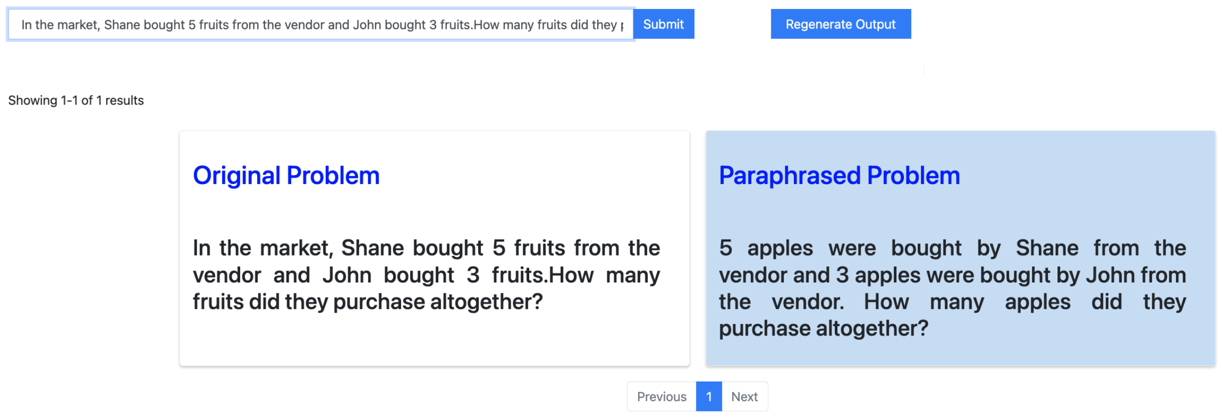
Table 6.10: Some sample outputs from `STEAD` and `PAP` .

original	noised	prompt	paraphrase
A grocer makes a 25% profit on the selling price for each bag of flour it sells. If he sells each bag for $100 and makes $3,000 in profit, how many bags did he sell?	A grocer on the selling cost for each pouch of tortillas it sold takes a 25% profit. Because he takes $ 3,000 in profit and sold two pouch as $ 100, how many pouch would he sold?	-	By selling a bag of flour a grocer makes a 25% profit. If he takes $3,000 in profit and sells each bag for $100, how many bags of flour would he sell?
A sales person gets a 10% commission on each sale he makes. How many sales of $250 each must he make in order to reach a salary of at least $1000?	Only selling person come. gets a 10% panel wednesday each selling he Of $ 250 each How many selling should he come in reorder to reach a pays of at least $ 1000?	-	A person is selling widgets. He gets a 10% commission on each widget he sells for $250 each. How many widgets must he sell in order to reach a price of at least $1000?
A grocery sells a bag of ice for $1.25, and makes 20% profit. If it sells 500 bags of ice, how much total profit does it make?	A NOUN1 sold a NOUN2 of NOUN3 as $ 1.25, also makes 20% profit. If it sold 500 NOUN2 which NOUN3, how much total profit accomplishes it make?	-	A piece of candy is sold by a store for $1.25, and a 20% profit is made by a retailer. If 500 pieces of candy were sold by the store, how much total profit did it make?
While selling a watch, a shopkeeper gives a discount of 5%. If he gives a discount of 7%, he earns Rs. 15 less as profit. The marked price of the watch is:	-	paraphrase passive replace :	A discount of 5% is given by a shopkeeper. Rs. 15 less is earned by him as profit than if he gives a discount of 7%. The marked price of a watch is:
Calculate the maximum distance you can travel with $8.50 on a cab which charges $3.50 for the first quarter-mile and 10 cents for each additional quarter mile.	-	paraphrase fix replace shuffle :	A taxi charges $3.50 for the first quarter-mile and 10 cents for each additional quarter mile. Calculate the maximum distance you can travel with $8.50.
In a bag of red and green sweets, the ratio of red sweets to green sweets is 3:4. If the bag contains 120 green sweets, how many red sweets are there?	-	paraphrase passive replace :	Ratio is 3:4 in a bag of red and green sweets. 120 green sweets are contained in the bag, how many red sweets are there?
If 27 bottles of soda cost A cents, how much will B bottles cost in dollars?	-	paraphrase fix :	A bottle of juice costs 27 cents, how much does B bottles cost in dollars?

6.6 Insights

- The proposed self-supervised paraphrasing framework for the new task of MWP paraphrasing provides diverse, fluent and faithful paraphrases. Here, faithfulness refers to retaining the solvability of the original problem without resolving numerical entities or units.

- The proposed self-supervised paraphrase quality detector *ParaQD* provides a clear separation between solvable and unsolvable paraphrases. This proves to be an useful metric when paraphrasing to filter out low quality outputs.


In the market, Shane bought 5 fruits from the vendor and John bought 3 fruits.How many fruits did they p

Submit

Regenerate Output

Showing 1-1 of 1 results

Original Problem

In the market, Shane bought 5 fruits from the vendor and John bought 3 fruits.How many fruits did they purchase altogether?

Paraphrased Problem

5 apples were bought by Shane from the vendor and 3 apples were bought by John from the vendor. How many apples did they purchase altogether?

Previous 1 Next

Figure 6.9: A screenshot of the deployed system. The input is entered into the query box, and the original problem as well as the generated paraphrase are displayed.

6.7 Discussions

SCANING has been deployed within a question generation system. The goal is to aid academicians in formulating diverse questions and provide more practice for the students. The students also benefit by reading and reasoning the steps to solve the problems rather than memorizing patterns to arrive at the solution. A screenshot of the system is depicted in Figure 6.9. We deploy PAP as it works in the input to paraphrase mapping space without directly depending upon noise. We also provide an option to regenerate the response, which generates a different response by leveraging diverse beam search. The deployment is also facilitated by the model-agnostic nature of our framework, as smaller quantized models are utilized for deployment to reduce latency.

A limitation of our proposed method is that the combined noising combinations we use may not be the global optimum. Is there a way we can determine the best noising combination to apply, given an input sample? We plan to further explore this question by involving meta-learning techniques to estimate the optimal noising combination. Another area for improvement is controlling the surface form of the output through prompting. Our prompting method allows for this but does not always return outputs faithful to the prompt. Perhaps stricter filtering thresholds and training on a larger corpus might help with this.

An exciting area to explore in more depth would be the interpolation of discrete prompts, where we could control the text in the desired manner by combining various prompts that separately signify the aspects we want to control, *without necessarily training on that exact combination of prompts*. We conducted some initial experiments regarding this idea with the proposed framework, and we will explore it in detail in the future, as currently, our method does not allow for guaranteed prompt interpolation.

In summary, our work proposes to build pipelines for automated content curation, tagging, and enrichment. This set of pipelines would aid the academicians in content curation and the students in achieving better learning outcomes. In the next chapter, we provide a brief summary of the work and potential directions to facilitate further research.

CHAPTER 7

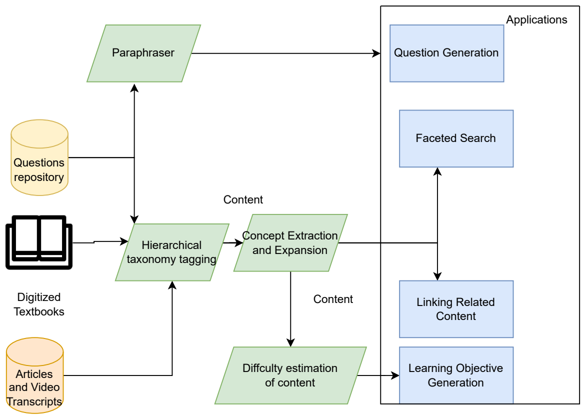
CONCLUSION

7.1 Summary

This chapter summarizes our work on automating content curation and enrichment in online learning platforms. The goal of the work was to build tools to aid academicians by focusing on proactive engagement in online learning platforms. Figure 7.1 demonstrates a summary of all the modules. We evaluate our platform on standard benchmarks and on data collected from our partner company Extramarks.

- The modules have been employed at scale to compile the *Authoring tool* at Extra-Marks. The tool envisions aiding academicians in curating content from diverse sources like digitized textbooks, video transcripts and third-party vendors. The tool is currently in its pilot version.

- The first three modules are also used in generating learning objectives, as demonstrated at the end of Chapter 5. The learning objectives are critical to tracking the progress of a student. Our tool obviates the need for manual generation and is also



Paraphraser
Applications
Question Generation
Faceted Search
Questions repository
Content
Hierarchical taxonomy tagging
Concept Extraction and Expansion
Digitized Textbooks
Content
Linking Related Content
Articles and Video Transcripts
Diffculty estimation of content
Learning Objective Generation

Figure 7.1: High level view of the proposed modules

pedagogically grounded as it primarily relies on of the difficulty level of the given learning content coupled with the concepts extracted from them.

– The taxonomy tagging module and concept expansion module have also been used to deploy other systems like *QDup* [42] for question de-duplication and related questions recommendation.

– The proposed modules have also been used in a system for unsupervised conversion of the objective to subjective questions to generate new questions [153].

– As part of our efforts to enhance the online learning experience, we also digitize NCERT books and collect relevant data for K-12 education from various sources [41]. We open-source this data on huggingface to facilitate learning science research. We also pre-trained BERT on this data to showcase the potential for a backbone model for applications related to education. We pose several sub-problems to tackle for the future in the K12BERT work.

7.2 Limitations and Future Work

The proposed systems have shown to be quantitatively better than existing related computational approaches. The resulting tools have also been deployed at our partner company ExtraMarks. However, the proposed approaches only deal with the proactive technological interventions [13] in online learning platforms. However, for a more holistic learning experience real-time user engagement and feedback should be considered for personalizing content to individual user needs. Appropriate technological interventions to predict student learning outcomes, per module performance and engagement with learning content [13] are some problems that deal with the reactive engagement of technology for learning. We do not deal with the reactive component in this dissertation. However, the proposed modules, like question difficulty estimation and content linking, could be extended to consider student engagement. The proposed modules can be extended with further scope in online learning platforms. Further limitations and possible extensions are as follows:

- The proposed approaches primarily deal with learning contents in English. However, to democratize access, the modules should also be extended to regional languages. This would ensure accessibility to learners and academicians across demographics.

- We do not explicitly deal with all possible biases in the proposed models and related data. For instance, the difficulty estimation module could be biased by the length of the phrasing of questions. A detailed analysis is necessary to establish possible data biases that could affect model performance and consequently the learning experience.

Apart from the limitations discussed, there are also several possible extensions to the proposed computational approaches for applicability in other domains.

- The efficient cross-attention mechanism proposed for *Tagrec++* can be used for dense passage retrieval to model the relatedness between a given query and candidate documents. It would help advance the performance of dense encoders by overcoming the quadratic complexity involved in attention [57].

- The taxonomy tagging framework currently operates in the euclidean space. However, the hyperbolic space is known to encode hierarchical data [154] naturally. A possible extension would be to extend the proposed dense retrieval approach to the hyperbolic space.

- The proposed self-supervised paraphrasing framework *SCANING* can be extended to general domain paraphrasing as the noising functions help in inducing syntactic and semantic changes. This could be particularly useful in generating diverse questions for other domains. It could also be useful to generate paraphrased versions of content where the style of the content could be steered using the generative model. For instance, a video transcript could be paraphrased to generate a reading comprehension exercise.

REFERENCES

[1] M. Syvyi, O. Mazbayev, O. Varakuta, N. Panteleeva, and O. Bondarenko, *Distance learning as innovation technology of school geographical education*, 2022. arXiv: 2202.08697 `[cs.CY]`.

[2] J. Kessels, "A relational approach to curriculum design," in *Design Approaches and Tools in Education and Training*, J. van den Akker, R. M. Branch, K. Gustafson, N. Nieveen, and T. Plomp, Eds. Dordrecht: Springer Netherlands, 1999, pp. 59–70.

[3] G. Heidari, A. Ramadan, M. Stocker, and S. Auer, "Demonstration of faceted search on scholarly knowledge graphs," in *Companion Proceedings of the Web Conference 2021*, ACM, Apr. 2021.

[4] F. St-Hilaire *et al.*, "A comparative study of learning outcomes for online learning platforms," in *Artificial Intelligence in Education*, I. Roll, D. McNamara, S. Sosnovsky, R. Luckin, and V. Dimitrova, Eds., Cham: Springer International Publishing, 2021, pp. 331–337.

[5] L. Benedetto *et al.*, "A survey on recent approaches to question difficulty estimation from text," 9, vol. 55, New York, NY, USA: Association for Computing Machinery, Jan. 2023.

[6] P. Pardjono, "Active learning: The dewey, piaget, vygotsky, and constructivist theory perspectives," *Jurnal Ilmu Pendidikan*, vol. 9, Feb. 2016.

[7] D. Xu *et al.*, "Multi-class hierarchical question classification for multiple choice science exams," in *Proceedings of the 12th Language Resources and Evaluation Conference*, Marseille, France: European Language Resources Association, May 2020, pp. 5370–5382.

[8] Z. Kozareva, "Everyone likes shopping! multi-class product categorization for e-commerce," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 1329–1333.

[9] N. V. Stash, A. I. Cristea, and P. M. De Bra, "Authoring of learning styles in adaptive hypermedia: Problems and solutions," in *Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers amp; Posters*, ser. WWW Alt. '04, New York, NY, USA: Association for Computing Machinery, 2004, pp. 114–123.

[10] L. P. Bradford, "The teaching-learning transaction," *Adult Education*, vol. 8, no. 3, pp. 135–145, 1958. eprint: https://doi.org/10.1177/074171365800800303.

[11] C. R. Rogers, “Personal thoughts on teaching and learning,” *Merrill-Palmer Quarterly (1954-1958)*, vol. 3, no. 4, pp. 241–243, 1957. (visited on 02/26/2023).

[12] K. Xue, V. Yaneva, C. Runyon, and P. Baldwin, “Predicting the difficulty and response time of multiple choice questions using transfer learning,” in *BEA*, ACL, Jul. 2020.

[13] S. Mallik and A. Gangopadhyay, “Proactive and reactive engagement of artificial intelligence methods for education: A review,” arXiv, 2023.

[14] D. Kolb, “Experiential learning: Experience as the source of learning and development,” vol. 1, Jan. 1984.

[15] A. N. Kumar, “Generation of problems, answers, grade, and feedback—case study of a fully automated tutor,” 3, vol. 5, New York, NY, USA: Association for Computing Machinery, Sep. 2005, 3–es.

[16] H. Chau, I. Labutov, K. Thaker, D. He, and P. Brusilovsky, “Automatic concept extraction for domain and student modeling in adaptive textbooks,” *International Journal of Artificial Intelligence in Education*, vol. 31, Aug. 2020.

[17] J. Gui, T. Chen, Q. Cao, Z. Sun, H. Luo, and D. Tao, “A survey of self-supervised learning from multiple perspectives: Algorithms, theory, applications and future trends,” arXiv, 2023.

[18] B. Min *et al.*, “Recent advances in natural language processing via large pre-trained language models: A survey,” arXiv, 2021.

[19] W. X. Zhao, J. Liu, R. Ren, and J.-R. Wen, “Dense text retrieval based on pretrained language models: A survey,” arXiv, 2022.

[20] J. Gou, B. Yu, S. J. Maybank, and D. Tao, “Knowledge distillation: A survey,” *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, Mar. 2021.

[21] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, “A survey of quantization methods for efficient neural network inference,” arXiv, 2021.

[22] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, “Efficient transformers: A survey,” *ACM Comput. Surv.*, vol. 55, no. 6, Dec. 2022.

[23] A. Patel, S. Bhattamishra, and N. Goyal, “Are NLP models really able to solve simple math word problems?” In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Lan-*

guage Technologies, Online: Association for Computational Linguistics, Jun. 2021, pp. 2080–2094.

[24] R. Gupta, V. Venktesh, M. Mohania, and V. Goyal, "'john ate 5 apples' != 'john ate some apples': Self-supervised paraphrase quality detection for algebraic word problems," in *Machine Learning and Knowledge Discovery in Databases*, M.-R. Amini, S. Canu, A. Fischer, T. Guns, P. Kralj Novak, and G. Tsoumakas, Eds., Cham: Springer Nature Switzerland, 2023, pp. 353–369.

[25] J. Zhou and S. Bhat, "Paraphrase generation: A survey of the state of the art," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 5075–5086.

[26] N. Ng, K. Cho, and M. Ghassemi, "SSMBA: Self-supervised manifold based data augmentation for improving out-of-domain robustness," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online: Association for Computational Linguistics, Nov. 2020, pp. 1268–1283.

[27] Y. Yang and X. Liu, "A re-examination of text categorization methods," in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '99, Berkeley, California, USA: Association for Computing Machinery, 1999, pp. 42–49.

[28] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Proceedings of the Fourteenth International Conference on Machine Learning*, ser. ICML '97, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, pp. 412–420.

[29] X. Qiu, J. Zhou, and X. Huang, "An effective feature selection method for text categorization," in *Proceedings of the 15th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining - Volume Part I*, ser. PAKDD'11, Shenzhen, China: Springer-Verlag, 2011, pp. 50–61.

[30] K. S. Hasan and V. Ng, "Automatic keyphrase extraction: A survey of the state of the art," *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1262–1273, Jun. 2014.

[31] A. Bougouin, F. Boudin, and B. Daille, "TopicRank: Graph-based topic ranking for keyphrase extraction," in *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, Nagoya, Japan: Asian Federation of Natural Language Processing, Oct. 2013, pp. 543–551.

[32] Z. Liu, W. Huang, Y. Zheng, and M. Sun, "Automatic keyphrase extraction via topic decomposition," in *Proceedings of the 2010 Conference on Empirical Meth-*

ods in Natural Language Processing, Cambridge, MA: Association for Computational Linguistics, Oct. 2010, pp. 366–376.

[33] Z. Liu, P. Li, Y. Zheng, and M. Sun, "Clustering to find exemplar terms for keyphrase extraction," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore: Association for Computational Linguistics, Aug. 2009, pp. 257–266.

[34] R. Mihalcea and P. Tarau, "TextRank: Bringing order into text," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 404–411.

[35] K. Bennani-Smires, C. Musat, A. Hossmann, M. Baeriswyl, and M. Jaggi, "Simple unsupervised keyphrase extraction using sentence embeddings," in *Proceedings of the 22nd Conference on Computational Natural Language Learning*, Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 221–229.

[36] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992.

[37] Y. He and D. Xin, "Seisa: Set expansion by iterative similarity aggregation," ser. WWW '11, New York, NY, USA: Association for Computing Machinery, 2011, pp. 427–436.

[38] S. Supraja, K. Hartman, S. Tatinati, and A. W. H. Khong, "Toward the automatic labeling of course questions for ensuring their alignment with learning outcomes," in *EDM*, 2017.

[39] U. Padó, "Question difficulty – how to estimate without norming, how to use for automated grading," in *BEA*, Copenhagen, Denmark: ACL, Sep. 2017.

[40] A. LW *et al.*, *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. Jan. 2001.

[41] V. Goel, D. Sahnan, V. Venktesh, G. Sharma, D. Dwivedi, and M. Mohania, "K-12bert: Bert for k-12 education," in *Artificial Intelligence in Education. Posters and Late Breaking Results, Workshops and Tutorials, Industry and Innovation Tracks, Practitioners' and Doctoral Consortium*, M. M. Rodrigo, N. Matsuda, A. I. Cristea, and V. Dimitrova, Eds., Cham: Springer International Publishing, 2022, pp. 595–598.

[42] M. Chowdhary, S. Goyal, V. V, M. Mohania, and V. Goyal, "Unsupervised question duplicate and related questions detection in e-learning platforms," in *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, ser. WSDM '23, Singapore, Singapore: Association for Computing Machinery, 2023, pp. 1200–1203.

[43] I. Arroyo, C. Beal, T. Murray, R. Walles, and B. P. Woolf, "Web-based intelligent multimedia tutoring for high stakes achievement tests," in *Intelligent Tutoring Systems*, J. C. Lester, R. M. Vicari, and F. Paraguaçu, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 468–477.

[44] C. Pande, H. F. Witschel, A. Martin, and D. Montecchiari, "Hybrid conversational ai for intelligent tutoring systems," CEUR Workshop Proceedings ,AAAI-MAKE 2021, Apr. 2021.

[45] F. St-Hilaire *et al.*, "A new era: Intelligent tutoring systems will transform online learning for millions," arXiv, 2022.

[46] J.-w. Ahn *et al.*, "Intelligent virtual reality tutoring system supporting open educational resource access," in *Intelligent Tutoring Systems*, R. Nkambou, R. Azevedo, and J. Vassileva, Eds., Cham: Springer International Publishing, 2018, pp. 280–286.

[47] C. Walkington, "Using adaptive learning technologies to personalize instruction to student interests: The impact of relevant contexts on performance and learning outcomes," *Journal of Educational Psychology*, vol. 105, p. 932, Nov. 2013.

[48] I. Perikos, F. Grivokostopoulou, K. Kovas, and I. Hatzilygeroudis, "Automatic estimation of exercises' difficulty levels in a tutoring system for teaching the conversion of natural language into first-order logic," *Expert Sys: J. Knowl. Eng.*, vol. 33, no. 6, pp. 569–580, Dec. 2016.

[49] R. Meng, S. Han, Y. Huang, D. He, and P. Brusilovsky, "Knowledge-based content linking for online textbooks," in *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, 2016, pp. 18–25.

[50] R. Meng, S. Zhao, S. Han, D. He, P. Brusilovsky, and Y. Chi, "Deep keyphrase generation," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 582–592.

[51] H. Chau, I. Labutov, K. Thaker, D. He, and P. Brusilovsky, "Automatic concept extraction for domain and student modeling in adaptive textbooks," *International Journal of Artificial Intelligence in Education*, vol. 31, Aug. 2020.

[52] W. Xia, W. Zhu, B. Liao, M. Chen, L. Cai, and L. Huang, “Novel architecture for long short-term memory used in question classification,” *Neurocomputing*, vol. 299, Mar. 2018.

[53] H.-F. Yu, C.-H. Ho, P. Arunachalam, M. Somaiya, and C.-J. Lin, “Product title classification versus text classification,” *Csie. Ntu. Edu. Tw*, pp. 1–25, 2012.

[54] S. Banerjee, C. Akkaya, F. Perez-Sorrosal, and K. Tsioutsiouliklis, “Hierarchical transfer learning for multi-label text classification,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 6295–6300.

[55] J. Wehrmann, R. C. Barros, S. N. d. Dôres, and R. Cerri, “Hierarchical multi-label classification with chained neural networks,” in *Proceedings of the Symposium on Applied Computing*, ser. SAC ’17, Marrakech, Morocco: Association for Computing Machinery, 2017, pp. 790–795.

[56] T. Lei, Z. Shi, D. Liu, L. Yang, and F. Zhu, “A novel cnn-based method for question classification in intelligent question answering,” in *Proceedings of the 2018 International Conference on Algorithms, Computing and Artificial Intelligence*, ser. ACAI 2018, Sanya, China: Association for Computing Machinery, 2018.

[57] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: pre-training of deep bidirectional transformers for language understanding,” *CoRR*, vol. abs/1810.04805, 2018. arXiv: 1810.04805.

[58] L. Tan, M. Y. Li, and S. Kok, “E-commerce product categorization via machine translation,” *ACM Trans. Manage. Inf. Syst.*, vol. 11, no. 3, 2020.

[59] K. Sinha, Y. Dong, J. C. K. Cheung, and D. Ruths, “A hierarchical neural attention-based text classifier,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 817–823.

[60] J. Zhou *et al.*, “Hierarchy-aware global model for hierarchical text classification,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 1106–1117.

[61] J. Lu, L. Du, M. Liu, and J. Dipnall, “Multi-label few/zero-shot learning with knowledge aggregated from multiple label graphs,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online: Association for Computational Linguistics, Nov. 2020, pp. 2935–2943.

[62] J. Fürnkranz, E. Hüllermeier, E. L. Mencía, and K. Brinker, “Multilabel classification via calibrated label ranking,” *Machine Learning*, vol. 73, pp. 133–153, 2008.

[63] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'13, Lake Tahoe, Nevada: Curran Associates Inc., 2013, pp. 3111–3119.

[64] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543.

[65] M. Pagliardini, P. Gupta, and M. Jaggi, "Unsupervised learning of sentence embeddings using compositional n-gram features," *arXiv preprint arXiv:1703.02507*, 2017.

[66] D. Cer *et al.*, "Universal sentence encoder for English," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 169–174.

[67] H. Bhathena, A. Willis, and N. Dass, "Evaluating compositionality of sentence representation models," in *Proceedings of the 5th Workshop on Representation Learning for NLP*, Online: Association for Computational Linguistics, Jul. 2020, pp. 185–193.

[68] A. Ettinger, A. Elgohary, C. Phillips, and P. Resnik, "Assessing composition in sentence vector representations," in *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 1790–1801.

[69] X. Wan and J. Xiao, "Single document keyphrase extraction using neighborhood knowledge," in *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, ser. AAAI'08, Chicago, Illinois: AAAI Press, 2008, pp. 855–860.

[70] R. Wang, W. Liu, and C. McDonald, "Corpus-independent generic keyphrase extraction using word embedding vectors," Software Engineering Research Conference (Vol 39), 2015.

[71] C. Florescu and C. Caragea, "Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1105–1115.

[72] F. Boudin, “Unsupervised keyphrase extraction with multipartite graphs,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018.

[73] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *J. Mach. Learn. Res.*, vol. 3, no. null, pp. 993–1022, Mar. 2003.

[74] M. Pagliardini, P. Gupta, and M. Jaggi, “Unsupervised learning of sentence embeddings using compositional n-gram features,” in *NAACL-HLT*, 2018.

[75] D. Mahata, J. Kuriakose, R. Shah, and R. Zimmermann, “Key2vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 2018, pp. 634–639.

[76] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, *Enriching word vectors with subword information*, 2017. arXiv: 1607.04606 `[cs.CL]`.

[77] V. Yaneva, C. Orăsan, R. Evans, and O. Rohanian, “Combining multiple corpora for readability assessment for people with cognitive disabilities,” in *BEA*, Sep. 2017.

[78] T. François and E. Miltsakaki, “Do NLP and machine learning improve traditional readability formulas?” In *Proceedings of the First Workshop on Predicting and Improving Text Readability for target reader populations*, Montréal, Canada: ACL, Jun. 2012.

[79] L. A. Ha and V. Yaneva, “Automatic distractor suggestion for multiple-choice tests using concept embeddings and information retrieval,” in *BEA*, Jun. 2018.

[80] T. Alsubait, B. Parsia, and U. Sattler, “A similarity-based theory of controlling mcq difficulty,” in *ICEEE 2013*, 2013, pp. 283–288.

[81] J. Liu, Q. Wang, C.-Y. Lin, and H.-W. Hon, “Question difficulty estimation in community question answering services,” in *Proceedings of the 2013 EMNLP*, Seattle, Washington, USA: ACL, Oct. 2013.

[82] Q. Wang, J. Liu, B. Wang, and L. Guo, “A regularized competition model for question difficulty estimation in community question answering services,” in *EMNLP 2014*, 2014, pp. 1115–1126.

[83] F. Nadeem and M. Ostendorf, “Language based mapping of science assessment items to skills,” in *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, Copenhagen, Denmark: ACL, Sep. 2017.

[84] M. Peters *et al.*, “Deep contextualized word representations,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 2227–2237.

[85] X. Zhang, J. Zhao, and Y. LeCun, “Character-level convolutional networks for text classification,” in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS’15, Montreal, Canada: MIT Press, 2015, pp. 649–657.

[86] Z. Xie *et al.*, “Data noising as smoothing in neural network language models,” in *International Conference on Learning Representations*, 2017.

[87] J. Wei and K. Zou, “EDA: Easy data augmentation techniques for boosting performance on text classification tasks,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 6382–6388.

[88] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, “Unsupervised data augmentation for consistency training,” in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS’20, Vancouver, BC, Canada: Curran Associates Inc., 2020.

[89] Z. Miao, Y. Li, and X. Wang, “Rotom: A meta-learned data augmentation framework for entity matching, data cleaning, text classification, and beyond,” in *Proceedings of the 2021 International Conference on Management of Data*. New York, NY, USA: Association for Computing Machinery, 2021, pp. 1303–1316.

[90] D. Kang, T. Khot, A. Sabharwal, and E. Hovy, “AdvEntuRe: Adversarial training for textual entailment with knowledge-guided examples,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 2418–2428.

[91] A. Asai and H. Hajishirzi, “Logic-guided data augmentation and regularization for consistent question answering,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 5642–5650.

[92] B. Dolan, C. Quirk, and C. Brockett, "Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources," in *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, Switzerland: COLING, Aug. 2004, pp. 350–356.

[93] S. Humeau, K. Shuster, M.-A. Lachaux, and J. Weston, "Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring," in *International Conference on Learning Representations*, 2020.

[94] N. Thakur, N. Reimers, J. Daxenberger, and I. Gurevych, "Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online: Association for Computational Linguistics, Jun. 2021, pp. 296–310.

[95] M. Ellsworth and A. Janin, "Mutaphrase: Paraphrasing with FrameNet," in *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, Prague: Association for Computational Linguistics, Jun. 2007, pp. 143–150.

[96] S. Narayan, S. Reddy, and S. B. Cohen, "Paraphrase generation from latent-variable PCFGs for semantic parsing," in *Proceedings of the 9th International Natural Language Generation conference*, Edinburgh, UK: Association for Computational Linguistics, Sep. 2016, pp. 153–162.

[97] C. Quirk, C. Brockett, and W. Dolan, "Monolingual machine translation for paraphrase generation," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 142–149.

[98] A. Prakash *et al.*, "Neural paraphrase generation with stacked residual LSTM networks," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, Osaka, Japan: The COLING 2016 Organizing Committee, Dec. 2016, pp. 2923–2934.

[99] S. Wang, R. Gupta, N. Chang, and J. Baldridge, "A task in a suit and a tie: Paraphrase generation with semantic augmentation," 01, vol. 33, Proceedings of the AAAI Conference on Artificial Intelligence, Jul. 2019, pp. 7176–7183.

[100] S. R. Bowman, L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, and S. Bengio, "Generating sentences from a continuous space," in *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 10–21.

[101] X. Zhang, Y. Yang, S. Yuan, D. Shen, and L. Carin, "Syntax-infused variational autoencoder for text generation," in *Proceedings of the 57th Annual Meeting of the*

Association for Computational Linguistics, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 2069–2078.

[102] N. Ng, K. Cho, and M. Ghassemi, "SSMBA: Self-supervised manifold based data augmentation for improving out-of-domain robustness," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online: Association for Computational Linguistics, Nov. 2020, pp. 1268–1283.

[103] X. Liu, L. Mou, F. Meng, H. Zhou, J. Zhou, and S. Song, "Unsupervised paraphrasing by simulated annealing," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 302–312.

[104] V. Viswanathan, M. Mohania, and V. Goyal, "Tagrec++: Hierarchical label aware attention network for question categorization," arXiv, 2022.

[105] A. Vyas, A. Katharopoulos, and F. Fleuret, "Fast transformers with clustered attention," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS'20, Vancouver, BC, Canada: Curran Associates Inc., 2020.

[106] B. Gao and L. Pavel, "On the properties of the softmax function with application in game theory and reinforcement learning," arXiv, 2017.

[107] A. Frome *et al.*, "Devise: A deep visual-semantic embedding model," in *Advances in neural information processing systems*, 2013, pp. 2121–2129.

[108] W. Lu, J. Jiao, and R. Zhang, "Twinbert: Distilling knowledge to twin-structured compressed bert models for large-scale retrieval," ser. CIKM '20, New York, NY, USA: Association for Computing Machinery, 2020, pp. 2645–2652.

[109] V. Venktesh, M. Mohania, and V. Goyal, "Tagrec: Automated tagging of questions with hierarchical learning taxonomy," in *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track*, Y. Dong, N. Kourtellis, B. Hammer, and J. A. Lozano, Eds., Cham: Springer International Publishing, 2021, pp. 381–396.

[110] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in neural information processing systems*, 2017, pp. 4077–4087.

[111] B. Chen, X. Huang, L. Xiao, Z. Cai, and L. Jing, "Hyperbolic interaction model for hierarchical multi-label classification," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, pp. 7496–7503, Apr. 2020.

[112] V. Bhatnagar, D. Kanojia, and K. Chebrolu, “Harnessing abstractive summarization for fact-checked claim detection,” in *Proceedings of the 29th International Conference on Computational Linguistics*, Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 2934–2945.

[113] Y. Wang *et al.*, “A neural corpus indexer for document retrieval,” in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 25 600–25 614.

[114] N. Craswell, “Mean reciprocal rank,” in *Encyclopedia of Database Systems*, L. LIU and M. T. ÖZSU, Eds. Boston, MA: Springer US, 2009, pp. 1703–1703.

[115] Y. Kim, M. Kim, A. Cattle, J. Otmakhova, S. Park, and H. Shin, “Applying graph-based keyword extraction to document retrieval,” in *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, Nagoya, Japan: Asian Federation of Natural Language Processing, Oct. 2013, pp. 864–868.

[116] S. Jones and M. S. Staveley, “Phrasier: A system for interactive document retrieval using keyphrases,” in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR ’99, Berkeley, California, USA: Association for Computing Machinery, 1999, pp. 160–167.

[117] V. Venktesh, M. Mohania, and V. Goyal, “Topic aware contextualized embeddings for high quality phrase extraction,” in *Advances in Information Retrieval*, M. Hagen *et al.*, Eds., Cham: Springer International Publishing, 2022, pp. 457–471.

[118] L. McInnes, J. Healy, N. Saul, and L. Großberger, “Umap: Uniform manifold approximation and projection,” *Journal of Open Source Software*, vol. 3, no. 29, p. 861, 2018.

[119] A. Hulth, “Improved automatic keyword extraction given more linguistic knowledge,” in *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, 2003, pp. 216–223.

[120] I. Augenstein, M. Das, S. Riedel, L. Vikraman, and A. McCallum, “SemEval 2017 task 10: ScienceIE - extracting keyphrases and relations from scientific publications,” in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, Vancouver, Canada: Association for Computational Linguistics, Aug. 2017.

[121] S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin, “SemEval-2010 task 5 : Automatic keyphrase extraction from scientific articles,” in *Proceedings of the 5th*

International Workshop on Semantic Evaluation, Uppsala, Sweden: Association for Computational Linguistics, Jul. 2010, pp. 21–26.

[122] V. Venktesh, M. S. Akhtar, M. Mohania, and V. Goyal, "Auxiliary task guided interactive attention model for question difficulty prediction," in *Artificial Intelligence in Education*, M. M. Rodrigo, N. Matsuda, A. I. Cristea, and V. Dimitrova, Eds., Cham: Springer International Publishing, 2022, pp. 477–489.

[123] A. Gogus, "Bloom's taxonomy of learning objectives," N. M. Seel, Ed., pp. 469–473, 2012.

[124] S. Ruder, "An overview of multi-task learning in deep neural networks," *arXiv preprint arXiv:1706.05098*, 2017.

[125] A. Vaswani *et al.*, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[126] N. A. Smith, M. Heilman, and R. Hwa, "Question generation as a competitive undergraduate course project," in *Proceedings of the NSF Workshop on the Question Generation Shared Task and Evaluation Challenge*, 2008, pp. 4–6.

[127] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 328–339.

[128] L. Benedetto, G. Aradelli, P. Cremonesi, A. Cappelli, A. Giussani, and R. Turrin, "On the application of transformers for estimating the difficulty of multiple-choice questions from text," in *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, Online: ACL, Apr. 2021, pp. 147–157.

[129] M. S. Akhtar, A. Ekbal, and E. Cambria, "How intense are you? predicting intensities of emotions and sentiments using stacked ensemble [application notes]," *IEEE Computational Intelligence Magazine*, vol. 15, no. 1, pp. 64–75, 2020.

[130] C. Hegde and S. Patil, "Unsupervised paraphrase generation using pre-trained language models," arXiv, 2020.

[131] R. Gupta, V. V., M. Mohania, and V. Goyal, "Coherence and diversity through noise: Self-supervised paraphrase generation via structure-aware denoising," arXiv, 2023.

[132] E. Egonmwan and Y. Chali, “Transformer and seq2seq model for paraphrase generation,” in *Proceedings of the 3rd Workshop on Neural Generation and Translation*, Hong Kong: Association for Computational Linguistics, Nov. 2019, pp. 249–255.

[133] T. Gao, X. Yao, and D. Chen, “SimCSE: Simple contrastive learning of sentence embeddings,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 6894–6910.

[134] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, “Pegasus: Pre-training with extracted gap-sentences for abstractive summarization,” in *Proceedings of the 37th International Conference on Machine Learning*, ser. ICML’20, JMLR.org, 2020.

[135] Y. Zhang, J. Baldridge, and L. He, “PAWS: Paraphrase adversaries from word scrambling,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 1298–1308.

[136] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, “Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers,” in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS’20, Vancouver, BC, Canada: Curran Associates Inc., 2020.

[137] Z. Jia, M. Maggioni, J. Smith, and D. P. Scarpazza, *Dissecting the nvidia turing t4 gpu via microbenchmarking*, 2019. arXiv: 1903.07486 `[cs.DC]`.

[138] Z. Jia, M. Maggioni, B. Staiger, and D. P. Scarpazza, *Dissecting the nvidia volta gpu architecture via microbenchmarking*, 2018. arXiv: 1804.06826 `[cs.DC]`.

[139] W. Ling, D. Yogatama, C. Dyer, and P. Blunsom, “Program induction by rationale generation: Learning to solve and explain algebraic word problems,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 158–167.

[140] C. Raffel *et al.*, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *J. Mach. Learn. Res.*, vol. 21, no. 1, Jan. 2020.

[141] N. Stiennon *et al.*, “Learning to summarize from human feedback,” in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS’20, Vancouver, BC, Canada: Curran Associates Inc., 2020.

[142] G. Yasui, Y. Tsuruoka, and M. Nagata, “Using semantic similarity as reward for reinforcement learning in sentence generation,” in *Proceedings of the 57th An-*

nual Meeting of the Association for Computational Linguistics: Student Research Workshop, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 400–406.

[143] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[144] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, “Language models are unsupervised multitask learners,” OpenAI, 2019.

[145] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, “Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing,” *ACM Comput. Surv.*, vol. 55, no. 9, Jan. 2023.

[146] M. Lewis *et al.*, “BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 7871–7880.

[147] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: A method for automatic evaluation of machine translation,” in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, Jul. 2002, pp. 311–318.

[148] T. Liu and D. W. Soh, “Towards better characterization of paraphrases,” in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 8592–8601.

[149] J. Carbonell and J. Goldstein, “The use of mmr, diversity-based reranking for reordering documents and producing summaries,” in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, 1998, pp. 335–336.

[150] J. Choquette, W. Gandhi, O. Giroux, N. Stam, and R. Krashinsky, “Nvidia a100 tensor core gpu: Performance and innovation,” *IEEE Micro*, vol. PP, pp. 1–1, Feb. 2021.

[151] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, “Unsupervised data augmentation for consistency training,” in *Proceedings of the 34th International Conference*

on Neural Information Processing Systems, ser. NIPS'20, Vancouver, BC, Canada: Curran Associates Inc., 2020.

[152] K.-H. Huang and K.-W. Chang, "Generating syntactically controlled paraphrases without using annotated parallel pairs," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, Online: Association for Computational Linguistics, Apr. 2021, pp. 1022–1033.

[153] A. Chhabra, N. Bansal, V. Venktesh, M. Mohania, and D. Dwivedi, "Obj2sub: Unsupervised conversion of objective to subjective questions," in *Artificial Intelligence in Education. Posters and Late Breaking Results, Workshops and Tutorials, Industry and Innovation Tracks, Practitioners' and Doctoral Consortium*, M. M. Rodrigo, N. Matsuda, A. I. Cristea, and V. Dimitrova, Eds., Cham: Springer International Publishing, 2022, pp. 467–470.

[154] B. Chen, X. Huang, L. Xiao, Z. Cai, and L. Jing, "Hyperbolic interaction model for hierarchical multi-label classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 7496–7503.