International Journal of Information Management Data Insights 1 (2021) 100025

Contents lists available at ScienceDirect

International Journal of Information Management Data Insights

journal homepage: www.elsevier.com/locate/jjimei

Optimization of paraphrase generation and identification using language models in natural language processing

Hemant Palivela[a,*]

Digital, eClerx Services, Mumbai, Maharashtra, India

ARTICLE INFO

Keywords:
Paraphrase identification
Paraphrase generation
Natural language generation
Language model
Encoder decoder
Transformer

ABSTRACT

Paraphrase Generation is one of the most important and challenging tasks in the field of Natural Language Generation. The paraphrasing techniques help to identify or to extract/generate phrases/sentences conveying the similar meaning. The paraphrasing task can be bifurcated into two sub-tasks namely, Paraphrase Identification (PI) and Paraphrase Generation (PG). Most of the existing proposed state-of-the-art systems have the potential to solve only one problem at a time. This paper proposes a light-weight unified model that can simultaneously classify whether given pair of sentences are paraphrases of each other and the model can also generate multiple paraphrases given an input sentence. Paraphrase Generation module aims to generate fluent and semantically similar paraphrases and the Paraphrase Identification system aims to classify whether sentences pair are paraphrases of each other or not. The proposed approach uses an amalgamation of data sampling or data variety with a granular fine-tuned Text-To-Text Transfer Transformer (T5) model. This paper proposes a unified approach which aims to solve the problems of Paraphrase Identification and generation by using carefully selected datapoints and a fine-tuned T5 model. The highlight of this study is that the same light-weight model trained by keeping the objective of Paraphrase Generation can also be used for solving the Paraphrase Identification task. Hence, the proposed system is light-weight in terms of the model's size along with the data used to train the model which facilitates the quick learning of the model without having to compromise with the results. The proposed system is then evaluated against the popular evaluation metrics like BLEU (BiLingual Evaluation Understudy):, ROUGE (Recall-Oriented Understudy for Gisting Evaluation), METEOR, WER (Word Error Rate), and GLEU (Google-BLEU) for Paraphrase Generation and classification metrics like accuracy, precision, recall and F1-score for Paraphrase Identification system. The proposed model achieves state-of-the-art results on both the tasks of Paraphrase Identification and paraphrase Generation.

1. Introduction

Natural Language Generation is a sub-field or a branch of Natural Language Processing (NLP). NLG is sometimes viewed closely with Natural Language Understanding (NLU) which is another sub-field of NLP. The human brain requires an in-depth understanding and thinking during the generation of language or sentences from scratch or by using a given context. It is more difficult for a computer system to generate meaningful and fluent sentences. Hence, for a system, NLG is more difficult than NLU. An ideal NLG system aims towards completely replacing humans for tasks like article writing, create summarizations/narratives quickly, real-time question-answering, report generations, and streamline operations. Some of the most important applications (Gandomi & Haider, 2015; He, Zha, & Li, 2013; Kumar, Kar, & Ilavarasan, 2021; Kushwaha, Kar, & Dwivedi, 2021) are Business Intelligence dashboard analysis, generation of financial portfolio summaries, plagiarism remover in academic papers, voice bots, and chatbots. The important phases in a typical NLG system are as follows:

1. Reason for Communication (Why to communicate?)
2. Content of the text (What exactly to say?)
3. How to formulate or say? (How to say it? - fluency, meaning)
4. Final Formulation

Natural Language Generation (NLG) can be viewed as a task of developing systems that can automatically write summaries, explanations, or narratives in either English or other languages. These NLG systems aim to generate or produce unambiguous and clear natural language just as the way people communicate with each other. Currently, there is a multitude of real-world applications where NLG can be used. These applications can range from chatbots (Kushwaha & Kar, 2020; Kushwaha, Kar, & Ilavarasan, 2020) or question answering systems (Harabagiu, Maiorano, & Pasca, 2003; Mollá & Vicedo, 2007; Voorhees, 2001) for an interac-

* Corresponding author.
E-mail address: hemant.datascience@outlook.com

https://doi.org/10.1016/j.jjimei.2021.100025
Received 23 May 2021; Received in revised form 27 June 2021; Accepted 28 June 2021

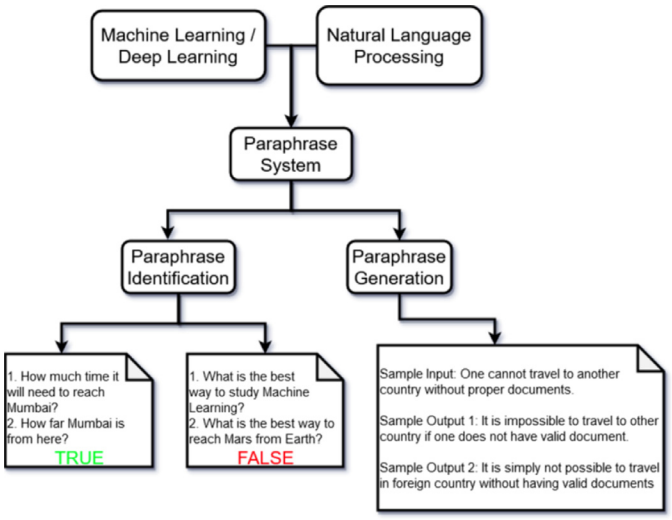

Fig. 1. Paraphrasing System.

tive textual communication or to generate weather reports or to caption images as well as generating human-like summaries (Cohn & Lapata, 2008; 2009; Galanis & Androutsopoulos, 2010) from research or academic papers, news articles and stories and also in machine translation systems (Koehn, 2009).

1. Mr. XYZ wrote a book on Artificial Intelligence
2. A book on Artificial Intelligence was written by Mr. XYZ
3. Mr. XYZ is an author of book related to Artificial Intelligence

These sentences convey almost the same meaning and hence they are the paraphrases of each other even though sentence 1 and 2 depicts that the book is completed but same cannot be said about sentence 3.

Fig. 1 depicts the general idea about the paraphrasing system. This paper presents a unified approach to solve both the sub-tasks of Paraphrase Generation and Identification using the same model. There has been a wide array of systems developed to solve the task of Paraphrase Identification and Paraphrase Generation. The Paraphrase Identification task is viewed as a supervised machine learning problem that can be solved by using traditional semantic similarly based techniques and with state-of-the-art deep learning algorithms like Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Long Short Term Memory (LSTM), etc. The Paraphrase Generation problem can be solved by using simple lexical features and word ordering or restructuring methods or by using templates extracted from WikiAnswers repositories. The most recent advancements in solving the task of Paraphrase Generation tasks involves using Generative Adversarial Networks (GANs), seq-to-seq based-models, and encoder-decoder based-models

This paper presents a unified system that combines the data selection variety parameter along with a custom fine-tuned T5 model especially for the Paraphrase Generation task which also solves the problem of Paraphrase Identification. An important feature of the T5 model is that it can be trained on multiple tasks simultaneously. The proposed system is trained on both the tasks of Paraphrase Identification and Paraphrase Generation tasks simultaneously and hence a lot of computational time and resources are saved as compared to training multiple models one at a time. This paper is structured by giving a detailed literature survey on Paraphrase Identification and Generation systems. Then, for both the tasks, a mathematical problem formulation is done and the proposed unified system architecture is discussed. This paper gives a thorough results analysis and concludes with a future scope and directions to improve.

One of the major challenges task research scholars face is to remove plagiarism in their respective articles. The proposed system aims to identify the plagiarized sentences and then automatically paraphrase those selected plagiarised sentences by using one unified model. The following are research aims to solve the following two research questions.

1. How can a system automatically highlight plagiarised sentences in a given article? (Paraphrase Identification)
2. How can a system rephrase the highlighted sentences from the previous step thereby removing plagiarism from the original sentence? (Paraphrase Generation)

This paper kick-starts with 2 where a formal mathematical formulation of both the problem statements (Paraphrase Identification and Paraphrase Generation) is explained. The Section 2 is followed by the Section 3 which describes in-detail the existing systems proposed to solve both the tasks. Further, the systems are classified into appropriate categories for better understanding. Then, the approached proposed in this paper is presented in the Section 4 along with the system architecture diagram. The proposed approach explanation in step-wise manner is followed by the Section 5 which explains various evaluation metrics along with reasons used to evaluate both the tasks in hand. The results obtained in this paper are then analysed, presented and compared against the latest state-of-the-art existing systems against different evaluation metrics in the Section 6. The discussions about the results and an in-depth contributions are stated in the Section 7. Finally, this paper concludes with the Section 8 which describes the conclusions and gives hints to research scholars about future scope.

2. Mathematical problem formulation

Paraphrasing can be sub-divided into two tasks namely Paraphrase Identification and Paraphrase Generation. The Paraphrase Identification task can be viewed as a discriminative type of task which tells whether a sentence pair points to the same meaning. In this task, the system might output a probability between 0 and 1 wherein the value tending to 1 depicts the sentence pair as a paraphrase of each other otherwise not. In some cases, the identification system outputs a semantic score which when normalized can help to discriminate between sentences pair. The Paraphrase Generation task aims to automatically generate one or multiple candidate paraphrases given the reference or input sentence. The aim is to generate semantically same and fluent paraphrases.

2.1. Paraphrase identification (PI) task

The PI task is viewed as a supervised machine learning task and is modeled as follows: Given a sentence pair (S_1, S_2), the aim is to find the target (1 or 0 which depicts the given sentence pair is paraphrase of each other or not respectively) where the sentence $S_1 = \{w_1, w_2, w_3, \ldots, w_n\}$ and $S_2 = \{w_1, w_2, w_3, \ldots, w_m\}$. It depicts that both the sentences length may vary. The output can be a probability between 0 and 1 or some normalized semantic scoring mechanism.

2.2. Paraphrase generation (PG) task

In the PI task, the aim is to generate a candidate sentence given an input sentence. Given an input sentence or a reference sentence S_1 where $S_1 = \{w_1, w_2, w_3, \ldots, w_n\}$, the aim is to generate one or more candidate sentences $S_2 = \{w_1, w_2, w_3, \ldots, w_m\}$, $S_3 = \{w_1, w_2, w_3, \ldots, w_o\}$, ... $S_4 = \{w_1, w_2, w_3, \ldots, w_p\}$. In this task too, the sentence length of the generated candidate sentences and the input or reference sentence may vary.

3. Review of literature

The paraphrase generation dates back to the year 1983 (McKeown, 1983). Quirk, Brockett, & Dolan (2004), Zhao, Niu, Zhou, Liu, & Li (2008) and Wubben, Van Den Bosch, & Krahmer (2010) attempted to solve the Paraphrase Generation task by using machine translation whereas (Bolshakov & Gelbukh, 2004)

and Kauchak & Barzilay (2006) proposed lexical based methods which generates paraphrases by words substitution. In recent times, due to advancements in the field of deep learning,]Gupta, Agarwal, Singh, & Rai (2018) and Fu, Feng, & Cunningham (2020) proposed neural network based methods.

The Paraphrase Generation techniques can be broadly classified in two major categories as follows:

1. Controlled Paraphrase Generation Methods The idea behind this approach is to generate paraphrases controlled by some templates or syntactic trees (Iyyer, Wieting, Gimpel, & Zettlemoyer, 2018) and Chen, Tang, Wiseman, & Gimpel (2019). An approach to generate paraphrases was proposed by Kumar, Ahuja, Vadapalli, & Talukdar (2020) which uses a mix of syntactic tree and tree encoder using Long Short Term Memory (LSTM) neural network. The main limitation is that it fails when the input dataset is noisy and grammatically not correct. A retriever-editor based approach was proposed by Kazemnejad, Salehi, & Baghshah (2020) to generate paraphrases. In this approach, a most similar source-target pair is selected by using embedding distance concerning the source. Then the role of the editor is to modify the input sentence by using a transformer. The retriever first selects the most similar source-target pair based on the embedding distance with the source. Then the editor modifies the input accordingly based on the Transformer (Vaswani et al., 2017). The limitation of this model is that it needs to train from scratch even though this model can restructure the sentence and introduce new words or can perform word substitution.
2. Pre-trained Language Models fine-tuning Large language models like GPT-2 (Radford et al., 2019) can be used to generate sentences in paraphrasing tasks. By using GPT-2, Witteveen & Andrews (2019) and Hegde & Patil (2020) proposed a paraphrase generation approach which exploits the capability of GPT-2 of understanding the language as the GPT-2 is generatively trained on the huge open-domain corpus. This approach aims to fine-tune the weights of the GPT-2 pretrained model. The major limitation is the source-copying in the output is observed. This limitation is taken care of in the proposed system (unified model) in this paper.

The paraphrase identification systems view the task of PI as a classification task and the paraphrase generation systems view the task of PG as a language generation task. Because of these totally different objectives, it is a major challenge to come up with a unified model capable of doing both the tasks as per the setup.

4. Unified system architecture

This paper proposes a unified system architecture capable of performing both the paraphrasing tasks of identification and generation. The following are the major system components:

1. Data Collection/Aquisition
2. Data Sampling Selection and Preprocessing
3. Text-to-text Transformer Hyperparameter Tuning
4. Text-to-text Transformer Training
5. Evaluation

1. Paraphrase Identification
2. Paraphrase Generation

4.1. Data collection/aquisition

The data is collected from different sources like PARANMT-50M (Wieting & Gimpel, 2017), Quora duplicate questions pair (Ansari & Sharma, 2020) and Microsoft Paraphrase Research Database (MSRP) (Dolan & Brockett, 2005). The ParaNMT database consists of more than 50 million sentential paraphrase pairs in the English language. To generate the huge ParaNMT corpus, a back-translation system was used. A Czech to English Neural Machine Translation (NLT) system was used to extract sentences written in Czech to English. The Quora duplicate questions pair dataset consists of a total of 404,290 sentence pairs. This data was split into 70-30 percent training and testing set. For training, the data consists of 283,003 sentence pairs and in test data, there are 121,287 sentence pairs. The MSRP dataset is filtered from a large sentence pairs database consisting of about 9,516,684 sentences. These sentences were extracted from news clusters spanning 2 years from World Wide Web (WWW). The final MSRP database consists of around 5800 sentence pairs. The training set consists of 4076 sentence pairs and 1725 sentence pairs in the test set. These three types of data are used to train the paraphrase model.

4.2. Data sampling selection and preprocessing

The objective is to promote data diversity by filtering and sampling the original data. It is observed that the paraphrase generation model outputs proper paraphrases without repetition by maximizing increasing the lexical, semantic, and syntactic diversity in the data used in training. This enables the paraphrasing model to generate more diverse paraphrases with the same meaning but having rich and varied vocabulary. The following transformations are applied to the training data to increase data diversity:

1. Remove the sentence pairs having more than 60% unigram, bi-gram or tri-gram overlap. This discourages the final trained model to copy the input sentence and maximizes the probability of generating diverse paraphrase.
2. Removing the sentence pairs having very less semantic similarly by using Sentence-BERT (Reimers & Gurevych, 2020). This forces the final trained model to generate semantically similar sentences.
3. In Quora and MSRP dataset, selecting only sentence pairs which are labelled as 1. (Here 1 denotes that the sentence pairs are paraphrases of each other)

By performing this step, the three main important parameters of diversity, semantically similar, and fluency are preserved. The diversity is preserved by restricting the copying. The semantically similar parameter is preserved by not allowing the model to train on those sentences which are not semantically similar. The final parameter of fluency is preserved by making sure the generated paraphrase is grammatically correct. This is automatically taken care of because, in all the training data sources, the sentence pairs are grammatically fluent. After applying all these filters, the data size shrunk to approximately 2 lakh sentence-pairs.

4.3. Paraphrase generation model building

The system is trained by keeping the objective of Paraphrase Generation. To train the system on the sentence pairs data, the Text-To-Text Transfer Transformer (Raffel et al., 2019) algorithm is used. In this study, a T5-base pretrained model is used which is then fine-tuned on the task of paraphrase generation. While fine-tuning for paraphrasing task, the hyper-parameters are also fine-tuned by using heuristics. The trained model is then used to generate paraphrases by specifying beam search and nucleus sampling, etc. The same trained model is used for Paraphrase Identification by extract sentence vector representations from the model.

4.4. Why text-To-Text transfer transformer?

Transfer learning has proved to be a very powerful technique in the field of Natural Language Processing (NLP). In transfer learning, an algorithm is first trained on a data-rich task (general/open or closed domain data) and then the trained model is finetuned on another downstream task. The Text-To-Text Transfer Transformer (T5) algorithm aims to convert every language problem into a text-to-text format. T5 is trained on a mix of labeled (Colossal Clean Crawled Corpus) and unlabelled data. The T5 model gives state-of-the-art results on more than 20 NLP

tasks. Hardly any technique performs consistently as T5 while preserving flexibility to train on any downstream task. In this paper, the T5-base model is used for fine-tuning and hyperparameter fine-tuning. This version of the T5 model contains approximately 220M parameters having 12-layers, 3072 feed-forwards hidden states, 768-hidden layers, and 12-heads.

4.5. Model architecture and mathematical representation

The T5 model used in this study is trained only for one task that is for text or paraphrase generation. The self-attention technique technique utilized in transformer takes input a sequence of input and generates a sequence of output which is of the same length that of input. In this case, the each element in the output sequence is computed by calculating a weighted average of the input sequence provided. Here, y^i depicts the i^{th} element in the output sequence and similarly x^j depicts the j^{th} element in the input sequence. Futher, each element y^i can be computed as $\sum_j w_{i,j} x_j$. $w_{i,j}$ is the weight which is to be optimized and this is the function of x^i and x^j. In this architecture, the encoder takes input the sequence of items (text tokens) and the decoder generates output sequence (text tokens).

The encoder used in this paper and depicted in the Fig. 3 utilizes a fully-visible attention mask. This type of fully-visible attention masking allows the self attention system to look into any of the entry of input when it simultaneously produces output in sequence by sequence fashion. As this model is used to generate paraphrases, a special prefix is added to the model which tells the model to generate text. Hence, this type of masking is appropriate when there is particular prefix supplied to the model. This prefix can also be said as providing a context to the model. In inference phase, this prefix is again used. The model architecture which is used in this paper is based on encoder-decoder component of transformer architecture. To give an overview about the model, an encoder generates a continuous representation of an input sequence. This representation (generated by encoder) represents all the learned information about the input sequence. The decoder takes the generated continuous representation step-wise and generates output per time-step (also previous step output is fed).

Further, to generate the sequences, a layer is defined under the decoder consisting of the following dual components:

1. Self-attention layer
2. Feed-forward layer

The sequences (paraphrases) are generated in autoregressive self-attention fashion which can be summarized as follows:

$$x_t^{(layer=l)} = LayerNormalization(z_t^{layer(l-1)}) \tag{1}$$

$$q_t^{(layer=l)}, k_t^{(layer=l)}, v_t^{(l)} = Qx_t^{(layer=l)}, Kx_t^{(l)}, Vx_t^{(l)} \tag{2}$$

$$h_t^{(layer=l)} = (h_{t-1}^{(layer=l)}, (k_t^{(layer=l)}, v_t^{(layer=l)})) \tag{3}$$

$$y_t^{(l)} = z_t^{(l-1)} + W_o Self - Attention - Layer(h_t^{(l)}, q_t^{(l)}) \tag{4}$$

Here, the Q, K, V, denote query, key and value weights or they are also called as projection matrices. For generating paraphrases of sentences having long term dependencies these query, key and value matrices play an important role. The query matrix comes from the decoder hidden state whereas the key and value comes from the encoder hidden states. The dot product between query and key denotes the compatibility score between key and query. Afterwards, these scores are inputted to the softmax function to yield a probability distribution (set of weights) of next token (summed to 1). Each of the weight value thus obtained is then multipled with its corresponding values to get a context vector which utlizes input hidden states. Finally, the output $y_t^{(layer=l)}$ is fed to a feed-forward neural network,

$$m_t^{(layer=l)} = LayerNormalization(y_t^{(layer=l)}) \tag{5}$$

$$z_t^{(layer=l)} = y_t^{(layer=l)} + W_2 Activation(W_1 m_t^{(layer=l)}) \tag{6}$$

As discussed, this output softmax layer outputs a probability distribution of the next token. While generating paraphrases, or during the inference phase, these output states are stored in order to get faster prediction in production mode and to avoid redundant computation.

The same model architecture is then used to identify whether sentence-pairs are paraphrases of each other or not. After the list of vectors are extracted, a cosine similarity is computed between vectors of sentence-pair and based on the cosine similarity score a decision is made about whether sentence-pair are paraphrases of each other or not. The cosine similarity is computed by using,

$$similarity(SV_1, SV_2) = \frac{SV_1 \cdot SV_2}{\|SV_1\| \times \|SV_2\|} \tag{7}$$

$$similarity(SV_1, SV_2) = \frac{\sum_{i=1}^{n} SV_{1i} \times SV_{2i}}{\sqrt{\sum_{i=1}^{n} SV_{1i}^2} \times \sqrt{\sum_{i=1}^{n} SV_{2i}^2}} \tag{8}$$

where, $SV_1 and SV_2$ are vectors of sentence 1 and sentence 2 respectively.

There are various available metrics used to compute similarity between two documents. One of the reason this study uses cosine similarity as a similarity metric because the vector representation of sentences differ in size and is highly dependent on sentence length. In such cases, the idea is to measure the angle between the two sentence vectors generated by the trained model projected in a multi-dimensional space. Another reason to use cosine similarity is because even if the euclidean distance between two sentence vectors is high there are chances that the same sentence vector pair can still be oriented closer together in reality. In cosine similarity, the smaller the angle, the higher will be the similarity between the two sentence vectors. Looking at a production perspective the cosine similarity is very fast and it also generates accurate results even if the data is sparse. Reimers & Gurevych (2019) also uses cosine similarity as a metric to compute semantic similarity between sentence vector pair.

The entire working of proposed model architecture is as follows:

1. Given an input sequence, the first step is to feed the sequence to word embedding layer. In this step, for each token, a vector representation is obtained.
2. A positional embedding is applied to the previous layer output. This step adds information about the positions to the input embeddings.
3. The output of previous step is then fed to the encoder layer which computes the continuous representation of input sequence. The following operations are computed such as: multi-headed attention (query, key, value vectors, dot product of query and key, scaling attention scores, softmax scaled scores, multiplying softmax with value vector)
4. The encoder's output is fed to decoder to generate output sequence. Most of the starting layers of decoder are similar to that of encoder witht he exception of the final softmax layer which outputs the probability distribution of the next token

4.6. Hyperparameter tuning of T5 model

It is not possible to fine-tune the T5 model by using the grid-search technique due to a high number of parameters and the model size. Hence, heuristic-based hyperparameter tuning was performed. The important parameters which were finetuned are learning rate, maximum sequence length, training batch size, and number of training epochs. The Table 1 depicts the hyperparameters used to train the T5 paraphrase generation model.

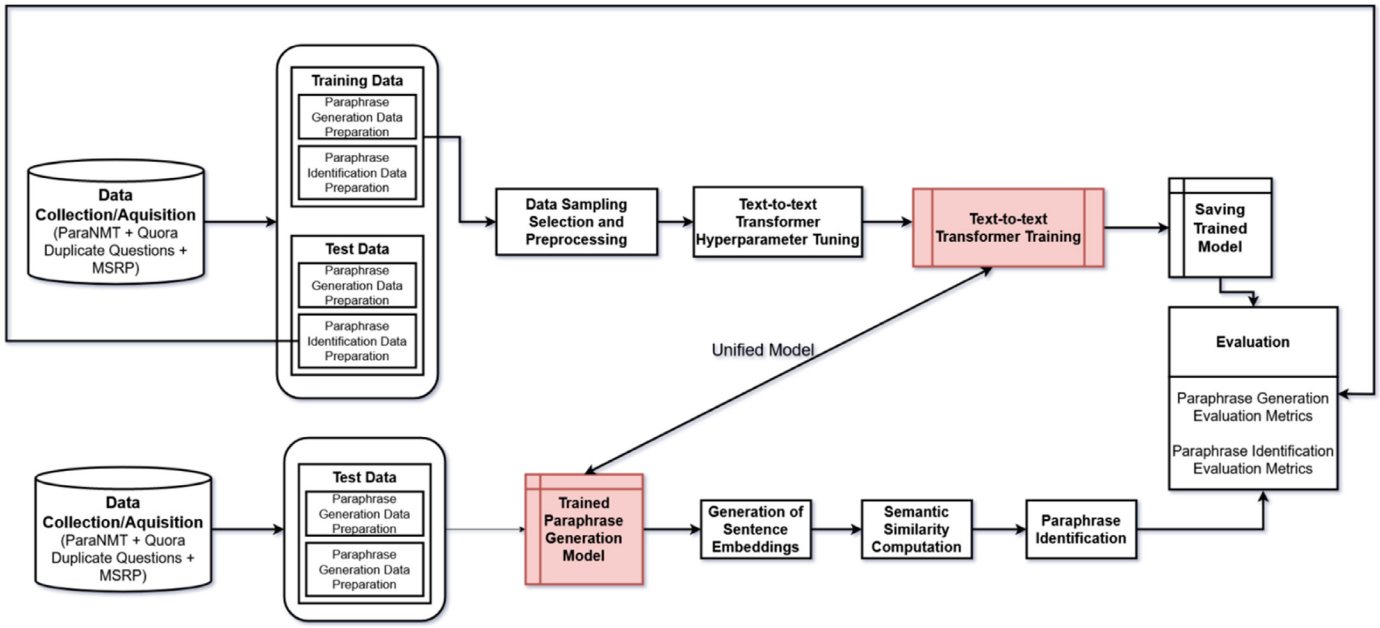

Fig. 2. Paraphrase Identification Process Flow.

Table 1
Final hyperparameters list used to finetune T5 Model for Paraphrase Generation and Paraphrase Identification Task.

Hyperparameter	Value
adam_epsilon	1e-08
best_model_dir	outputs/bestmodel
cache_dir	cache_dir/
cosine_schedule_num_cycles	0.5
do_lower_case	False
dynamic_quantize	False
early_stopping_consider_epochs	False
early_stopping_metric	eval_loss
early_stopping_metric_minimize	True
early_stopping_patience	2
adafactor_eps	(1e-30, 0.001)
adafactor_clip_threshold	1.0
adafactor_decay_rate	-0.8
adafactor_scale_parameter	False
adafactor_relative_step	False
adafactor_warmup_init	False
eval_batch_size	8
evaluate_during_training	False
evaluate_during_training_silent	True
evaluate_during_training_steps	2000
evaluate_during_training_verbose	False
evaluate_each_epoch	True

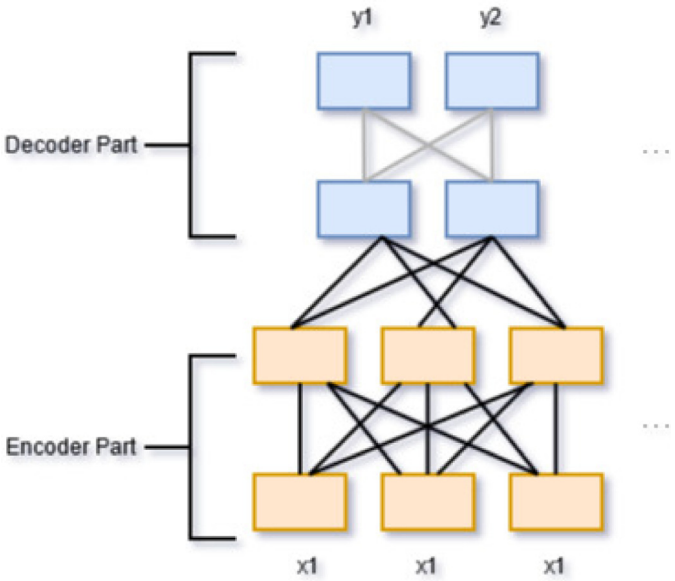

Fig. 3. Encoder Decoder Model Schematic.

4.7. Paraphrase identification

The main highlight of this paper is using the same model for Paraphrase Generation and Paraphrase Identification task. The same hyperparameter fine-tuned T5 model is utilized to solve the problem of Paraphrase Identification. The aim is to extract sentence vectors from sentence pairs and then computing semantic similarity between sentence pairs. Fig. 2 depicts the Paraphrase Identification workflow. The figure. The semantic similarity between sentence pairs is computed as follows:

1. Initially, load the trained T5 model on paraphrase Generation task.
2. Tokenize the sentence pair and extract token ids and then add padded tokens to it.
3. Extraction of attention mask and segment tokens.
4. With the help of token ids, attention mask, and segment token, compute the sentence vectors. In this step, we need to compute the pooled sentence representations which represent the embeddings for each word in the sentence.

4.8. Training time and system configuration

The entire model was trained for 200 epochs on a system with 120GB of RAM. The GPU used was A-100-PCIE having RAM of 40GB. The algorithm took 74 hours to train on Paraphrase Generation Task. The proposed system is lightweight and efficient and it can also be deployed into an actual production environment. By optimizing the parameters in beam search and sampling parameters, the performance can be further improved during the generation phase. Currently, the system can utilize multiple GPU's for multiple paraphrase generation.

5. Evaluation metrics

In this paper, separate evaluation metrics are used for Paraphrase Generation and Paraphrase Identification task. The following are the metrics used for the Paraphrase Identification task:

1. Accuracy: Before going into accuracy, precision, and F1-score, the following terminologies are important concerning the Paraphrase Identification task.
   - True Positives (TP): Model predicts sentence-pair is a paraphrase of each other and in reality, it is so.
   - True Negatives (TN): Model predicts sentence-pair is not a paraphrase of each other and in reality too, they are not paraphrases of each other.
   - False Positives (FP): Model predicts sentence-pair is a paraphrase of each other, but they are not paraphrases of each other (Type I error)
   - False Negatives (FN): Model predicts sentence-pair is not a paraphrase of each other, but they are paraphrases of each other. (Type II error)

   Accuracy is the fraction of predictions our model got right.
2. Precision: Precision is the value denoting by how often the model is correct if the model predicts that particular sentence-pairs are paraphrases of each other. Precision can be denoted by using: TP/predicted(Yes)
3. Recall: Recall tells when sentence-pairs are actually paraphrases of each other, how often does the model predict Yes. Recall can also be called sensitivity or True Positive rate. It is denoted as TP/actual(Yes).
4. F1-score: F1-score is the harmonic mean of precision and recall (True Positive Rate).

The Paraphrase Generation task is evaluated by using the following metrics:

1. ROUGE (Recall-Oriented Understudy for Gisting Evaluation): ROUGE is based only on recall and it is one of the most common metrics used for summarization tasks. But it can also be used to evaluate paraphrases. It has various types like ROUGE-1, ROUGE-2, ROUGE-N, and ROUGE-(L/W/S) depending on the feature. ROUGE-N is based on the number of grams. If unigram is set, then ROUGE-1 computes the recall by analyzing the matching unigrams. ROUGE-L/W/S denotes the ROUGE on Longest Common Subsequence (LCS), Weighted LCS, and skip-bigram co-occurrence statistics respectively. In this paper, ROUGE-1, ROUGE-2, and ROUGE-L are used.
2. BLEU (BiLingual Evaluation Understudy): BLEU counts the matching N-grams in the generated translation to N-grams in the gold or reference text. Here the unigram is token-wise and the bi-gram is word pair. To penalize a generated translation or paraphrase which generates a lot of reasonable words, the n-gram counting is modified. In this paper, BLUE-1/2/3/4 is used for evaluation.
3. GLEU (Google-BLEU): GLEU is a variant of BLEU score and it is aimed towards more evaluation close to human judgments. GLUE overcomes the BLUE's drawback of per sentence reward objective. GLUE works by computing n-gram precisions over the gold/truth/reference paraphrases but here the more weight is assigned to N-grams which have been changed from the source.
4. WER (Word Error Rate) WER is one of the most common metrics used in Automatic Speech Recognition (ASR) but can also be used to evaluate Paraphrase Generation. The WER can be summarized by, Word Error Rate = (Substitutions + Insertions + Deletions) / Number of Words Spoken. Where substitutions denote a replacement of a word, insertion denotes a word is added and deletions identify the words which are deleted.
5. METEOR (Metric for Evaluation of Translation with Explicit ORdering) METEOR works by modifying the precision and recall computations. It replaces them with a weighted F-Score. This F-score is based on mapping 1-grams or unigrams along with a penalty function whenever an incorrect word order is encountered.

Table 2
Evaluation results for Paraphrase Generation Task.

Evaluation Metrics	Test Datasets		
	ParaNMT	MSRP	Quora
ROUGE-1 Precision	0.5236	0.6350	0.6396
ROUGE-1 Recall	0.5443	0.5066	0.6296
ROUGE-1 F1-Score	0.5257	0.5524	0.6286
ROUGE-2 Precision	0.3579	0.4827	0.5448
ROUGE-2 Recall	0.3706	0.3867	0.5412
ROUGE-2 F1-Score	0.3591	0.4204	0.5409
ROUGE-3 Precision	0.5074	0.6154	0.6346
ROUGE-3 Recall	0.5172	0.4951	0.6251
ROUGE-3 F1-Score	0.5060	0.5387	0.6247
METEOR	0.4801	0.4971	0.6143
BLEU	0.2933	0.3235	0.5033
BLEU-1	0.5454	0.4830	0.6428
BLEU-2	3.70E-01	0.3716	0.5431
BLEU-3	3.01E-01	0.3167	0.5089
BLEU-4	0.2664	0.2817	0.4933
GLEU	0.5818	0.5742	0.6740
WER	0.7261	0.6150	0.6835

Table 3
Evaluation Results for Paraphrase Identification Task.

Dataset (threshold = 0.726)	Accuracy	Precision	Recall
MSRP	82.05	73.68	87.5
Quora	87.17	78.9	93.75

Table 4
Comparative Analysis for Paraphrase Generation for Quora Dataset.

Work by	ROUGE-1	ROUGE-2
Seq2Seq Li et al. (2017)	58.77	31.47
Residual LSTM Li et al. (2017)	59.21	32.43
Pointer-generator Li et al. (2017)	61.93	36.07
RL-ROUGE Li et al. (2017)	63.35	37.33
RbM-SL Li et al. (2017)	64.39	38.11
RbM-IRL Li et al. (2017)	64.02	37.72
Unified Approach (ours)	**62.8682**	**54.0932**

6. Results and comparative analysis

The Paraphrase Generation and Paraphrase Identification are evaluated on the same model but by using different evaluation metrics as the former is a generation task and the latter is viewed as a classification task. The Table 2 depicts the evaluation results for the Paraphrase Generation task. The evaluation is carried on three test datasets of ParaNMT, MSRP, and Quora on evaluation metrics of ROUGE-1, ROUGE-2, ROUGE-L, METEOR, BLEU, BLEU-1, BLEU-2, BLEU-3, BLEU-4, GLEU, and WER. The Table 3 represents the evaluation results for the Paraphrase Identification task. The evaluation for the Paraphrase Identification task was carried on MSRP and Quora datasets by using evaluation metrics of accuracy, precision, recall, and F1-score. To evaluate the PI task, a threshold of 0.726 was set. This threshold denotes that the sentence-pair will be assigned as 1(are paraphrases of each other) if the semantic score(computed from the unified model) is greater than the threshold (0.726) else the prediction will be assigned as 0. After trial and error, this threshold provided a good balance in precision, recall, and f1-score for both the classes for both test datasets. It can be seen that the unified trained model worked pretty well in both the tasks of Paraphrase Generation and Identification.

Further, a comparative analysis was performed for both the tasks with existing available systems for different datasets. The Table 4 and

Table 5
Comparative Analysis for Paraphrase Generation for MSRP Dataset.

Method	BLEU
Transfer Learning (Brad & Rebedea, 2017)	12.91
ParaSCI (Dong, Wan, & Cao, 2021)	27.18
Unified Approach (proposed)	**32.35**

Table 6
Comparative Analysis for Paraphrase Identification for MSRP Dataset.

Work by	Accuracy%
Accuracy (in percentage)	
Mihalcea et al. Mihalcea, Corley, Strapparava et al. (2006)	65.4
Kozareva and Montoyo Kozareva & Montoyo (2006)	76.6
Hassan Hassan (2011)	68.8
Hu et al. ARC-I Hu, Lu, Li, & Chen (2015)	69.6
Hu et al. ARC-II Hu et al. (2015)	69.9
Rus et al. Rus, McCarthy, Lintean, McNamara, & Graesser (2008)	70.6
Islam and Inkpen Islam & Inkpen (2009)	72.6
Yin et al. Yin & Schütze (2015)	72.5
Fernando and Stevenson Fernando & Stevenson (2008)	74.1
Wan et al. Wan, Dras, Dale, & Paris (2006)	75.6
Pang et al. Pang et al. (2016)	75.94
Socher et al. Socher, Huang, Pennington, Ng, & Manning (2011)	76.8
Madnani et al. Madnani, Tetreault, & Chodorow (2012)	77.4
Zhang et al. Zhang, Rong, Liu, Tian, & Xiong (2017)	77.5
Word2vector + Hybrid Deep Learning Kubal & Nimkar (2018)	77.66
GloVe + Hybrid Deep Learning Kubal & Nimkar (2018)	78.49
Context2vec + Hybrid Deep Learning Kubal & Nimkar (2018)	79.88
DeepPairwiseWord Lan et al. (2017)	83.4
Unified Approach (Proposed)	**82.0513**

Table 7
Comparative Analysis for Paraphrase Identification for Quora Dataset.

Work by	Accuracy (in percentage)
TF-IDF CatBoost Chandra & Stefanus (2020)	75.39
pt-DECATT Tomar, Duque, Täckström, Uszkoreit, & Das (2017)	88.4
BERT Corbeil & Ghadivel (2020)	83.5
XLNet Corbeil & Ghadivel (2020)	86
RoBERTa Corbeil & Ghadivel (2020)	88.6
ALBERT Corbeil & Ghadivel (2020)	86.7
Unified Approach (Proposed)	**87.17948**

Table 5 represents the comparative analysis performed for the Paraphrase Generation task for Quora and MSRP datasets respectively. It can be observed that the proposed system achieved higher scores as compared to existing systems. The proposed system achieved a ROUGE-1 score of 0.628, ROUGE2-score of 0.54, BLEU score of 0.5037, and METEOR score of 0.6143. The RbM-IRL (Li, Jiang, Shang, & Li, 2017) from Table 4 achieved the best ROUGE-1 score of 64.02 but the proposed system surpassed it and the remaining systems in other metrics by a huge margin.

The Paraphrase Identification system is compared with the existing systems from the year 2006. The tables 6 and 7 depicts the comparisons based on accuracy for the task of the Paraphrase Identification system. In Table 5, it can be seen that the proposed system achieves an accuracy of 82.0513% which surpasses the other existing systems except for the Deep Pairwise Word (Lan, Qiu, He, & Xu, 2017) which achieves 83.4% accuracy. The unified model is trained by keeping the objective of generation but still, the proposed approach computes a respectable accuracy level.

7. Discussion and contributions

Wahle, Ruas, Meuschke, & Gipp (2021) and!Wahle et al. (2021) and other language model based techniques mentioned in Section 3 solved the task of paraphrase identification using Language models like BERT (Devlin, Chang, Lee, & Toutanova, 2018), RoBERTa (Liu et al., 2019), etc. by viewing the task as traditional sentence-pair classification. In sentence-pair classification, the language model like BERT or RoBERTa (mostly a pretrained one) is finetuned on a dataset consisting of sentence pairs and their label. Here when the label is 1, then the sentence-pairs are paraphrases of each other otherwise not. This model architecture setup requires the pretrained language model to be fine-tuned separately for sentence-pair classification task. The limitation of such setup is repeated training and evaluation of the model when new data needs to be added. The proposed system solves this problem by preventing the model to be trained separately for sentence-pair classification task. Hence saving the resources in terms of time and computing resources as these language model's take a considerable amount of resources like Graphical Processing Unit (GPU), more primary memory and time required for training.

As discussed in Section 6 and looking at the comparative analysis between different systems, the proposed system is thoroughly evaluated against different metrics and outperforms the major state-of-the-art systems. The highlight is the classification model derived from the model trained to generate paraphrases also performs descent. It shows that the sentence vectors generated by the model have contextual information. While generating paraphrases, it is seen that the output sentences are diverse and not just a copy of input sentences. This diversity in output but simultaneously keeping the meaning intact is made possible by the data sampling step which ensures that there is variety in input and output. This makes the proposed system useful in paraphrasing sentences having technical jargon or words. The proposed system thus provides a single unified fine-tuned model capable of performing both the task of paraphrase identification and paraphrase generation. The way in which the data is generated in Section 4.2 proved to an important step that discouraged copying the input sentence as output. Further, the unique approach in the way the text-to-text transformer model is trained to generate sentences with fine-tuned parameters improved the quality of paraphrases generated by the model. Finally, the novel setup in which the trained model is used for a classification task (PI) performed decently to identify whether sentence pairs are paraphrases of each other. This system can further be extended to paraphrase actual academic text consisting of technical domain-related words. In this scenario, keeping track of these vital words would be of utmost importance as it is not appropriate to change these words (as these words can be species names, animals, monuments, drugs names, etc).

The contributions of this study is three-fold namely, the novel data sampling technique, paraphrase generation model training strategy and a unified model which is capable of solving the tasks of paraphrase generation as well as paraphrase identification. This study can also be extended to actually implement in production environment. The data sampling technique ensures reduction of sentence copying as the output sentence from the trained model. The training strategy makes sure that the model is accurate enough to generate multiple paraphrases for a given input sentence keeping the semantic meaning intact. The concept of unified modelling facilitates the usage of the same model to also use as to solve the paraphrase identification or paraphrase detection problem.

8. Conclusion and future scope

The paraphrasing tasks of Paraphrase Generation and Identification are the most important in the field of Natural Language Processing. Until now, both tasks can be solved by using different individual approaches. The Paraphrase Generation can be solved by using conditional generation based neural networks and the task of Paraphrase Identification can be solved by viewing it as a sentence-pair binary classification task. This paper proposed a model that can perform both tasks by training a single model with the objective of generation. The final trained model achieved state-of-the-art results on both tasks while surpassing the major existing systems as well in their corresponding evaluation metrics.

The generated paraphrases are not only grammatically fluent but there is negligible copying from the input sentence. The data sampling selection phase proved to be effective to avoid the output getting copied from the input. The proposed system further became strong by proper hyper-parameter tuning and the strengths of the T5 model for the text-to-text task of paraphrase generation. Further, due to the efficient and accurate sequential embeddings generation from the trained model and proper threshold value, the Paraphrase Identification task achieved respectable results. The proposed approach is designed in such a way that the end-to-end system can be used in a real-time production environment as the system takes advantage of multi-GPU training and parallel evaluations. The final model is evaluated on both the tasks for different test databases and then compared the individual tasks on independent test datasets with the existing system to give an idea about the performance of the proposed system. In this paper, the T5-base model is fine-tuned but it can be extended to T5-large and other variants too.

Declaration of competing interest

The author declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Ansari, N., & Sharma, R. (2020). Identifying semantically duplicate questions using data science approach: A quora case study. *arXiv preprint arXiv:2004.11694*.

Bolshakov, I. A., & Gelbukh, A. (2004). Synonymous paraphrasing using wordnet and internet. In *International conference on application of natural language to information systems* (pp. 312–323). Springer.

Brad, F., & Rebedea, T. (2017). Neural paraphrase generation using transfer learning. In *Proceedings of the 10th international conference on natural language generation* (pp. 257–261).

Chandra, A., & Stefanus, R. (2020). Experiments on paraphrase identification using quora question pairs dataset. *arXiv preprint arXiv:2006.02648*.

Chen, M., Tang, Q., Wiseman, S., & Gimpel, K. (2019). Controllable paraphrase generation with a syntactic exemplar. *arXiv preprint arXiv:1906.00565*.

Cohn, T., & Lapata, M. (2008). Sentence compression beyond word deletion. In *Proceedings of the 22nd international conference on computational linguistics (coling 2008)* (pp. 137–144).

Cohn, T. A., & Lapata, M. (2009). Sentence compression as tree transduction. *Journal of Artificial Intelligence Research, 34*, 637–674.

Corbeil, J.-P., & Ghadivel, H. A. (2020). Bet: A backtranslation approach for easy data augmentation in transformer-based paraphrase identification context. *arXiv preprint arXiv:2009.12452*.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dolan, W. B., & Brockett, C. (2005). Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the third international workshop on paraphrasing (iwp2005)*.

Dong, Q., Wan, X., & Cao, Y. (2021). Parasci: A large scientific paraphrase dataset for longer paraphrase generation. *arXiv preprint arXiv:2101.08382*.

Fernando, S., & Stevenson, M. (2008). A semantic similarity approach to paraphrase detection. In *Proceedings of the 11th annual research colloquium of the uk special interest group for computational linguistics* (pp. 45–52).

Fu, Y., Feng, Y., & Cunningham, J. P. (2020). Paraphrase generation with latent bag of words. *arXiv preprint arXiv:2001.01941*.

Galanis, D., & Androutsopoulos, I. (2010). An extractive supervised two-stage method for sentence compression. In *Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics* (pp. 885–893).

Gandomi, A., & Haider, M. (2015). Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management, 35*(2), 137–144.

Gupta, A., Agarwal, A., Singh, P., & Rai, P. (2018). A deep generative framework for paraphrase generation. In *Proceedings of the aaai conference on artificial intelligence: vol. 32*.

Harabagiu, S. M., Maiorano, S. J., & Pasca, M. A. (2003). Open-domain textual question answering techniques. *Natural Language Engineering, 9*(3), 231.

Hassan, S. (2011). *Measuring semantic relatedness using salient encyclopedic concepts*. University of North Texas.

He, W., Zha, S., & Li, L. (2013). Social media competitive analysis and text mining: A case study in the pizza industry. *International Journal of Information Management, 33*(3), 464–472.

Hegde, C., & Patil, S. (2020). Unsupervised paraphrase generation using pre-trained language models. *arXiv preprint arXiv:2006.05477*.

Hu, B., Lu, Z., Li, H., & Chen, Q. (2015). Convolutional neural network architectures for matching natural language sentences. *arXiv preprint arXiv:1503.03244*.

Islam, A., & Inkpen, D. (2009). Semantic similarity of short texts. *Recent Advances in Natural Language Processing V, 309*, 227–236.

Iyyer, M., Wieting, J., Gimpel, K., & Zettlemoyer, L. (2018). Adversarial example generation with syntactically controlled paraphrase networks. *arXiv preprint arXiv:1804.06059*.

Kauchak, D., & Barzilay, R. (2006). Paraphrasing for automatic evaluation. In *Proceedings of the human language technology conference of the naacl, main conference* (pp. 455–462).

Kazemnejad, A., Salehi, M., & Baghshah, M. S. (2020). Paraphrase generation by learning how to edit from samples. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 6010–6021).

Koehn, P. (2009). *Statistical machine translation*. Cambridge University Press.

Kozareva, Z., & Montoyo, A. (2006). Paraphrase identification on the basis of supervised machine learning techniques. In *International conference on natural language processing (in finland)* (pp. 524–533). Springer.

Kubal, D. R., & Nimkar, A. V. (2018). A hybrid deep learning architecture for paraphrase identification. In *2018 9th international conference on computing, communication and networking technologies (icccnt)* (pp. 1–6). IEEE.

Kumar, A., Ahuja, K., Vadapalli, R., & Talukdar, P. (2020). Syntax-guided controlled generation of paraphrases. *Transactions of the Association for Computational Linguistics, 8*, 330–345.

Kumar, S., Kar, A. K., & Ilavarasan, P. V. (2021). Applications of text mining in services management: A systematic literature review. *International Journal of Information Management Data Insights, 1*(1), 100008.

Kushwaha, A. K., & Kar, A. K. (2020). Language model-driven chatbot for business to address marketing and selection of products. In *International working conference on transfer and diffusion of it* (pp. 16–28). Springer.

Kushwaha, A. K., Kar, A. K., & Dwivedi, Y. K. (2021). Applications of big data in emerging management disciplines: A literature review using text mining. *International Journal of Information Management Data Insights, 1*(2), 100017.

Kushwaha, A. K., Kar, A. K., & Ilavarasan, P. V. (2020). Predicting information diffusion on twitter a deep learning neural network model using custom weighted word features. In *Conference on e-business, e-services and e-society* (pp. 456–468). Springer.

Lan, W., Qiu, S., He, H., & Xu, W. (2017). A continuously growing dataset of sentential paraphrases. *arXiv preprint arXiv:1708.00391*.

Li, Z., Jiang, X., Shang, L., & Li, H. (2017). Paraphrase generation with deep reinforcement learning. *arXiv preprint arXiv:1711.00279*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Madnani, N., Tetreault, J., & Chodorow, M. (2012). Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 182–190).

McKeown, K. (1983). Paraphrasing questions using given and new information. *American Journal of Computational Linguistics, 9*(1), 1–10.

Mihalcea, R., Corley, C., Strapparava, C., et al. (2006). Corpus-based and knowledge-based measures of text semantic similarity. In *Aaai: vol. 6* (pp. 775–780).

Mollá, D., & Vicedo, J. L. (2007). Question answering in restricted domains: An overview. *Computational Linguistics, 33*(1), 41–61.

Pang, L., Lan, Y., Guo, J., Xu, J., Wan, S., & Cheng, X. (2016). Text matching as image recognition. In *Proceedings of the aaai conference on artificial intelligence: vol. 30*.

Quirk, C., Brockett, C., & Dolan, W. B. (2004). Monolingual machine translation for paraphrase generation. In *Proceedings of the 2004 conference on empirical methods in natural language processing* (pp. 142–149).

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog, 1*(8), 9.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... Liu, P. J. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 conference on empirical methods in natural language processing*. Association for Computational Linguistics.

Reimers, N., & Gurevych, I. (2020). Making monolingual sentence embeddings multilingual using knowledge distillation. *arXiv preprint arXiv:2004.09813*.

Rus, V., McCarthy, P. M., Lintean, M. C., McNamara, D. S., & Graesser, A. C. (2008). Paraphrase identification with lexico-syntactic graph subsumption.. In *Flairs conference* (pp. 201–206).

Socher, R., Huang, E. H., Pennington, J., Ng, A. Y., & Manning, C. D. (2011). Dynamic pooling and unfolding recursive autoencoders for paraphrase detection.. In *Nips: vol. 24* (pp. 801–809).

Tomar, G. S., Duque, T., Täckström, O., Uszkoreit, J., & Das, D. (2017). Neural paraphrase identification of questions with noisy pretraining. *arXiv preprint arXiv:1704.04565*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Voorhees, E. M. (2001). The trec question answering track. *Natural Language Engineering, 7*(4), 361.

Wahle, J. P., Ruas, T., Meuschke, N., & Gipp, B. (2021). Are neural language models good plagiarists? a benchmark for neural paraphrase detection. *arXiv preprint arXiv:2103.12450*.

Wan, S., Dras, M., Dale, R., & Paris, C. (2006). Using dependency-based features to take thepara-farceout of paraphrase. In *Proceedings of the australasian language technology workshop 2006* (pp. 131–138).

Wieting, J., & Gimpel, K. (2017). Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. *arXiv preprint arXiv:1711.05732*.

Witteveen, S., & Andrews, M. (2019). Paraphrasing with large language models. *arXiv preprint arXiv:1911.09661*.

Wubben, S., Van Den Bosch, A., & Krahmer, E. (2010). Paraphrase generation as mono-

lingual translation: Data and evaluation. In *Proceedings of the 6th international natural language generation conference*.

Yin, W., & Schütze, H. (2015). Convolutional neural network for paraphrase identification. In *Proceedings of the 2015 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 901–911).

Zhang, X., Rong, W., Liu, J., Tian, C., & Xiong, Z. (2017). Convolution neural network based syntactic and semantic aware paraphrase identification. In *2017 international joint conference on neural networks (ijcnn)* (pp. 2158–2163). IEEE.

Zhao, S., Niu, C., Zhou, M., Liu, T., & Li, S. (2008). Combining multiple resources to improve smt-based paraphrasing model. In *Proceedings of acl-08: Hlt* (pp. 1021–1029).