

# CLEF 2023 SimpleText Tasks 2 and 3: Enhancing Language Comprehension: Addressing Difficult Concepts and Simplifying Scientific Texts Using GPT, BLOOM, KeyBert, Simple T5 and More

Petra Dadić<sup>1</sup>, Olga Popova<sup>2</sup>

<sup>1</sup>University of Split, 33 Ruđera Boškovića St., Split, 21000, Croatia

<sup>2</sup>University of Cadiz, 9 Paseo Carlos III. St., Cadiz, 11003, Spain

## Abstract

This paper addresses two tasks aimed at enhancing language comprehension. First, we address the problem of identifying complexity in the text. This task consists of two parts: identifying complex terms and explaining them. To accomplish this task, we used a variety of methods. Different approaches are compared using both manual and automated evaluation. The second task deals with text simplification. GPT, Bloom and Simple T5 are compared on a variety of automated metrics. After analyzing performance of different models, we concluded that it is possible to achieve decent results with most of the methods, even when the free and trial versions of the products are used.

## Keywords

CLEF, Simplification, Automatic Simplification

## 1. Introduction

Understanding and going through a lot of scientific literature is often crucial in domain understanding and quality research. However, understanding scientific texts can be a challenging task. They are often dense, and the researchers are usually not able to keep up, especially in the era where most of the research is interdisciplinary.

In this paper, we address two interconnected tasks aimed at solving this problem [1] [2]. Task 2 focuses on the identification and explanation of difficult concepts, aiming to help understanding specialized terminology and technical jargon. By automatically identifying difficult words, deciphering abbreviations, and explaining them in context, this project tries to help researchers understand scientific jargon that falls out of their area of expertise.

Task 3, on the other hand, revolves around the rewriting of scientific text to simplify complex information. Using natural language processing and machine learning, we hope to simplify passages from scientific abstracts in order to help identify the relevant articles and keep up with the ever-growing library of knowledge accessible through the Internet.

---


CLEF 2023: Conference and Labs of the Evaluation Forum, September 18-21, 2023, Thessaloniki, Greece

✉ pdadic@pmfst.hr (P. Dadić); olga.popova@uca.es (O. Popova)

ORCID 0000-0001-7084-3140 (O. Popova)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

The outcomes of our research have numerous implications across various domains. By improving language comprehension, we can enhance education and learning experiences, enable more effective information retrieval, and contribute to the standardization of terminology. Moreover, our work holds significance for natural language processing applications such as machine translation and question-answering systems, where accurate understanding and explanation of difficult terms are critical.

In the subsequent sections of this paper, we delve into the methodologies employed for both Task 2 and Task 3. We present our dataset, describe the techniques utilized, and provide experimental results and evaluations.

## **2. Field Overview**

### **2.1. Complexity Spotting**

There are different approaches to complexity spotting. Some of the more popular ones are listed below:

**Linguistic Features Approach:** Traditional approaches use linguistic features to determine text complexity. These features include word length, sentence length, syntactic complexity (e.g., number of clauses), lexical diversity, and readability formulas (e.g., Flesch-Kincaid readability score)[3]. Machine learning algorithms, such as decision trees or support vector machines, can be trained on labeled datasets using these features to predict text complexity.

**Domain-Specific Features Approach:** Scientific texts often have domain-specific characteristics that contribute to their complexity. These features can be identified using rule-based or statistical methods. [4]

**Corpus-Based Approaches:** Corpus-based methods use large-scale text corpora to analyze and model text complexity. These approaches usually measure the frequency and distribution of complex linguistic patterns. They often use statistical models and clustering to predict complexity.[5]

**Neural Networks Approach:** Deep learning models: recurrent neural networks (RNNs) and transformers can be trained on labeled datasets to use linguistic and contextual features to predict complexity. They have shown very good results as they can assess the difficulty of the terms in context.

Previously mentioned methods are further explained in this review article. [6]

**Ensemble Approaches:** Ensemble methods combine multiple models or techniques to improve prediction accuracy.

**Cross-Domain Transfer Learning Approach:** Transfer learning techniques, such as pre-trained language models like BERT[7] or GPT[8], are by far the most popular approach nowadays. They have been trained on large amounts of data and can be used to accomplish almost any NLP task. They show the best results with a bit of fine-tuning.

### **2.2. Text Simplification**

**Neural Machine Translation (NMT)[9]:** Models trained on parallel corpora consisting of complex and simplified sentences. They learn to generate simplified versions of input sentences by

leveraging the alignment between the two languages.

Seq2Seq Models with Attention Mechanism[10]: Sequence-to-sequence (Seq2Seq) models equipped with attention mechanisms capture dependencies between words in the source sentence and generate simplified output sentences more accurately.

Transformer Models: Transformer-based architectures, such as the Transformer model introduced in the "Attention is All You Need" paper[11], have been applied to text simplification tasks. These models use self-attention mechanisms to capture contextual information and generate simplified sentences.

Rule-Based Approaches: Rule-based methods incorporate predefined simplification rules and patterns to transform complex text. These rules can be based on linguistic principles or domain-specific guidelines. Rule-based approaches are often used in combination with other methods to enhance the performance of the simplification models.

### **3. Data Description**

Data used for tasks was sent to us by CLEF[1] as a part of our shared tasks. Data was different for each of the tasks, and it consisted of 3 test sets: small, medium and large. The small dataset is included in the medium one, while the latter is included in the large one.

For the first task, train data consists of 2 documents. The first is the Citation Network Dataset. The second is Qrels. This file extends the qrels released with a significant increase of the depth of judgments of abstracts per query. Relevance annotations are provided on a 0-2 scale (the higher, the more relevant) for 29 queries associated with the first 15 articles from the Guardian, totalling 203 examples. The test data consisted of 152072 for large dataset, for medium dataset 4797, and 2234 for small dataset.

For the third task, train data is a parallel corpus of 648 manually simplified sentences. Test data consists of 152072 in large dataset, 4976 in medium dataset, and 2413 in small dataset.

### **4. Method Description**

We used a variety of methods for each of the tasks. In this section, we will provide a short summary for all methods used.

#### **4.1. Difficult concept identification**

The goal of this sub-task is to identify the words that might be difficult for people to understand and extract them from the given sentences. Since there are not a lot of papers written on this topic, we decided to use methods used for keyword extraction as there is a strong correlation between keywords and difficult words in the sentences. It is important to note that the definition of difficult words was not provided, so we had to work on what seemed to be logical for us. For us, difficulty often meant scientific jargon and domain specific terms.

#### 4.1.1. KeyBERT

KeyBERT is a minimal and easy-to-use keyword extraction technique that leverages BERT embeddings to create keywords and keyphrases that are most similar to a document.[12]. Model used was 'all-MiniLM-L6-v2' with English stop words excluded.

#### 4.1.2. RAKE

Rapid Automatic Keyword Extraction (RAKE) is an algorithm to automatically extract keywords from documents. [13]. Implementation used was from rake nltk package for python[14].

#### 4.1.3. YAKE

YAKE is a unsupervised statistic based model for keyword extraction implemented based on the following paper[15].

#### 4.1.4. Bloom

BigScience Large Open-science Open-access Multilingual Language Model (BLOOM) is a transformer-based large language model.[16]. Prompt in appendix.

#### 4.1.5. Simple T5

Simple T5 is a model that enables quickly and simply training T5 models. Some of its features are reduced compared to T5.[17]. For this task, sentences were labeled as 'source text' and terms as 'target'

#### 4.1.6. TextRank

TextRank is a graph based model for keyword extraction, implemented based on paper by Mihalcea and Tarau.[18]. Implementation used the pke package for python.[19]

### 4.2. Difficult concept scoring

For this sub-task we used data derived from different approaches mentioned in the first sub-task. This section highlights some of the methods used in scoring. We had to use manual interventions to adjust the scores, as the project required the score to be between 0 and 2. Since, as mentioned before, we were given just the vague definitions of difficulty, our adjustment of the results was very subjective.

#### 4.2.1. The Flesch Reading Ease formula

The Flesch Reading Ease formula is one of the formulas used for assessing reading-ease, higher scores are calculated for material that is easier to read; lower numbers are calculated for texts that are more difficult to read.[20]

$$206.835 - 1.015 * \left( \frac{\text{total words}}{\text{total sentences}} \right) - 84.6 * \left( \frac{\text{total syllables}}{\text{total words}} \right) \quad (1)$$

#### 4.2.2. Flesch–Kincaid grade level

Flesch–Kincaid grade level is one of the formulas used for assessing reading-ease, scores indicate the grade a person would have to be in US education system to understand the text[3].

$$0.39 * \left(\frac{\text{total words}}{\text{total sentences}}\right) + 11.8 * \left(\frac{\text{total syllables}}{\text{total words}} - 15.59\right) \quad (2)$$

#### 4.2.3. Coleman–Liau index

Similar to Flesch-Kincaid grade level, Coleman–Liau index calculates the grade a person would have to be to comprehend the text. It uses the number of letters instead of the number of syllables[21].

$$CLI = 0.0588 * avg_{Letters \text{ per } 100words} - 0.296 * avg_{Sentences \text{ per } 100words} - 15.8 \quad (3)$$

#### 4.2.4. Automated readability index

Readability test for English text, also uses the US education system as a scale[22].

$$4.71 * \left(\frac{\text{characters}}{\text{words}}\right) + 0.5 * \left(\frac{\text{words}}{\text{sentences}}\right) - 21.43 \quad (4)$$

#### 4.2.5. Dale-Chall Readability Score

Method that uses the lookup table of 3000 words and calculates the grade based on the formula[23]:

$$0.1579 * \left(\frac{\text{difficult words}}{\text{words}} * 100\right) + 0.0496 * \left(\frac{\text{words}}{\text{sentences}}\right) \quad (5)$$

### 4.3. Task 2.2: Difficult Term Explanation

The goal of this task was to provide explanation to the terms extracted and scored in the previous tasks. The methods we used in explanations are listed below.

#### 4.3.1. Bloom

BigScience Large Open-science Open-access Multilingual Language Model (BLOOM) is a transformer-based large language model.[16]

#### 4.3.2. WordNet

WordNet[24] is large English language dictionary that can be used in python as a part of NLTK library[14].

#### 4.3.3. PyDict

PyDictionary is a dictionary module for Python.

#### **4.3.4. Wikipedia**

Wikipedia is a Python library that enables users to search and parse Wikipedia pages.

#### **4.4. Task 3**

The goal of the third task was to simplify complicated sentences. We only used 3 models: Bloom, GPT and SimpleT5. As we were relying on our own accounts and had to save on tokens we only used 100 examples with GPT and Bloom.

##### **4.4.1. Bloom**

BigScience Large Open-science Open-access Multilingual Language Model (BLOOM) is a transformer-based large language model.[16]

##### **4.4.2. GPT**

GPT is a pretrained transformer large scale language learning model[8]

##### **4.4.3. SimpleT5**

Simple T5 is model that enables quickly and simply training T5 models. Some of its features are reduced compared to T5.[17]

### **5. Limitations**

This project was done as a collaboration between two students as a part of Erasmus+ Blended Intensive Program. Both of us are fairly new in the field of Natural Language Processing and due to equipment limitations we couldn't implement any of the more complex models. We were using our own laptops and the free version of Google Collaboratory environment, which limits the data allocation and session duration for its users. We used a free version of the models and had to keep track of tokens in order to complete all the tasks, and thus we decided to use a reduced number of examples to work with BLOOM and GPT. Keeping that in mind, we tried to represent as many different approaches so, our project would give a good comparison for models that can be implemented with limited resources.

### **6. Results**

#### **6.1. Difficult words identification and scoring**

For these tasks, our results were scored by selecting a smaller subset of all submitted difficult terms. Afterward the selected terms were scored on three metrics: number of ok terms, number of correctly scored terms and number of terms that satisfy both criteria.

### 6.1.1. Number of ok terms

Number of correct terms reflect how many terms are correctly identified as difficult.

As a team, for this task we submitted 10 runs using a combination of methods. Number of extracted, evaluated and ok terms are shown in a table below.

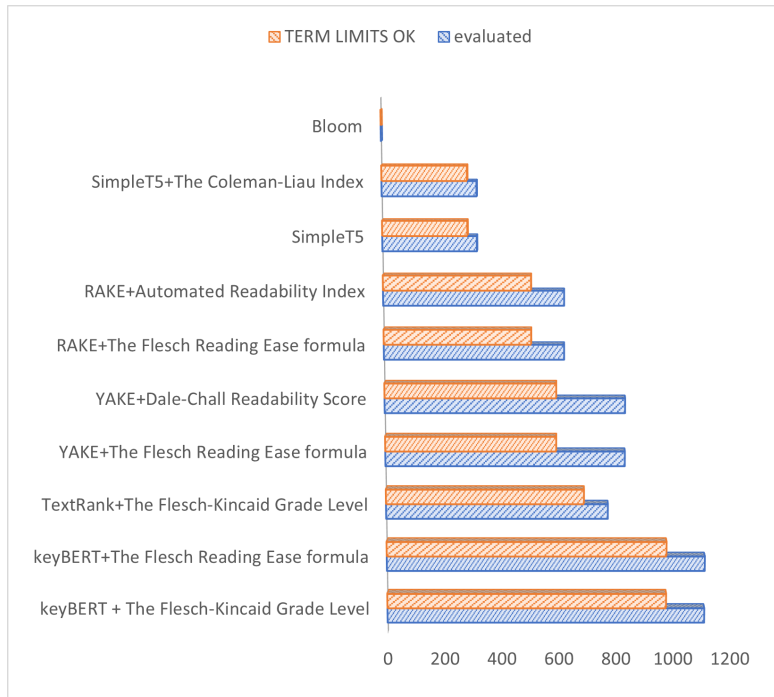
Table 1: The table showing the number of all extracted terms, evaluated terms and the number of correct terms.

Extraction Method	Total Terms	Evaluated Terms	Terms OK
keyBERT + The Flesch-Kincaid Grade Level	11099	1109	975
keyBERT+The Flesch Reading Ease formula	11099	1109	975
TextRank+The Flesch-Kincaid Grade Level	10056	770	687
YAKE+The Flesch Reading Ease formula	11112	828	591
YAKE+Dale-Chall Readability Score	11112	828	591
RAKE+The Flesch Reading Ease formula	10660	618	505
RAKE+Automated Readability Index	10660	618	505
SimpleT5	2234	321	289
SimpleT5+The Coleman-Liau Index	2234	321	289
Bloom	100	0	0

As is visible from the table above different methods returned different number of words so, it is best to take into the account how many of the actual terms were ok. The relation between number of extracted and correct terms is shown below.

Table 2: The table showing percentage of correct terms.

Extraction Method	Number of correct terms
Bloom	0
SimpleT5+The Coleman-Liau Index	90%
SimpleT5	90,0%
RAKE+Automated Readability Index	81,7%
RAKE+The Flesch Reading Ease formula	81,7%
YAKE+Dale-Chall Readability Score	71,3%
YAKE+The Flesch Reading Ease formula	71,3%
TextRank+The Flesch-Kincaid Grade Level	89,2%
keyBERT+The Flesch Reading Ease formula	87,9%
keyBERT + The Flesch-Kincaid Grade Level	87,9%



**Figure 1:** Relationship between the number of all evaluated and correct terms

A lot of the methods achieved good results. Bloom had the worst results and Simple T5 had the best, scoring 90%. This is somewhat surprising as we expected Bloom to achieve better results. However, this could also be explained with the number of total terms that we submitted, which were only 100 for Bloom and way more for other methods (1109 for keyBERT). The reason for this was the token limitation that forced us to use less examples with Bloom. Since there were fewer examples to evaluate, it seems like none of the examples matched the ones provided by Bloom.

### 6.1.2. Number of correctly scored terms

The number of correctly scored terms reflects if the complexity of the term is adequately labeled. Scoring the terms proved a bit more challenging as it was dependent on the results of the previous task.

As is shown in the table below, scoring was quite poor across all methods. This is probably due to the fact that the train set consisted of only 203 examples and didn't include any examples of class 0. The Task description stated that participants should score terms "... on a scale 0-2 (2 to be the most difficult terms, while the meaning of terms scored 0 can be derived or guessed)" so we had to adjust the scores to reflect a different scoring system.

### 6.1.3. Correct LIMITS & Correct difficulty scores

Correct LIMITS & Correct difficulty scores reflects how many terms satisfy both criteria.



Table 3: The table showing the number of correctly identified terms, number of correctly scored and terms that satisfy both criteria.

Extraction Method	Identified	Scored	Both	% both
Bloom	0	0	0	0
SimpleT5+The Coleman-Liau Index	289	51	44	15,22%
SimpleT5	289	87	77	26,6%
RAKE+Automated Readability Index	505	184	157	31,1%
RAKE+The Flesch Reading Ease formula	505	216	186	36,8%
YAKE+Dale-Chall Readability Score	591	187	126	21,3%
YAKE+The Flesch Reading Ease formula	591	302	229	38,7%
TextRank+The Flesch-Kincaid Grade Level	687	198	179	26,1%
keyBERT+The Flesch Reading Ease formula	975	298	262	26,9%
keyBERT + The Flesch-Kincaid Grade Level	975	374	352	36,1%

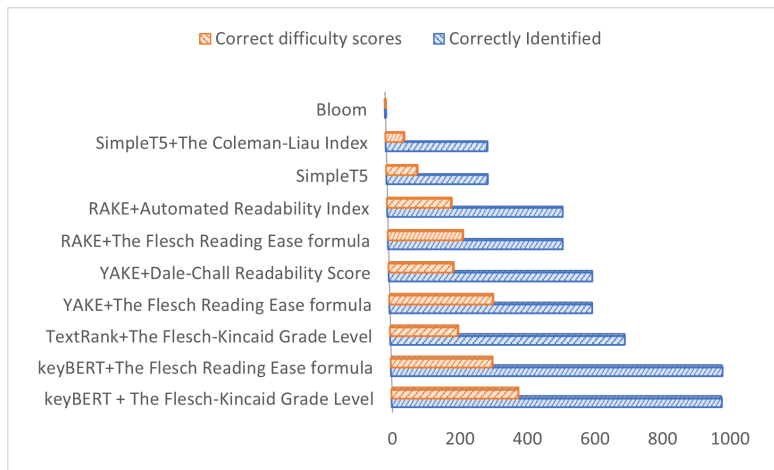


Figure 2: Relationship between the number of all correctly identified terms and correctly scored terms

All the methods required manual intervention, which could be another source of error. Our scores were adjusted by looking at the output of each of the methods and manually deciding on where to draw a line of most difficult terms.

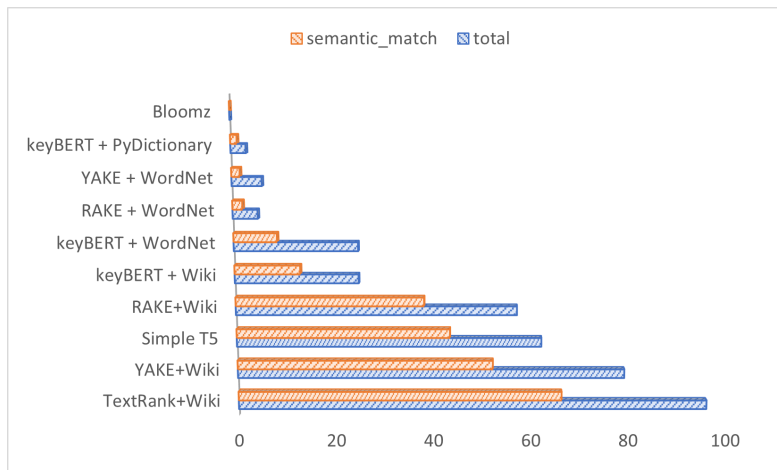
The best results were achieved using Flesch Reading Ease formula paired with RAKE and YAKE extraction methods.

## 6.2. Difficult Term Explanation

For this task our results were calculated using a combination of BLEU [25], ROUGE[26] and matching scores.

### 6.2.1. Matching scores

Matching scores are scores that evaluate how many definitions are exactly, partially or semantically equivalent to target definitions. Terms that were identified in the previous tasks were explained in this task using a variety of methods. The following figure shows the number of evaluated explanations that semantically matched target definitions. This task was done on 3 sets of data. Smaller subset of test set, extended version of the test set and non-unique words that also appeared in the train data set.



**Figure 3:** Relationship between the number evaluated definitions and number of sentences that semantically match the target in the smaller subset.

Table 4: The table showing the percentage of correctly explained terms in smaller subset.

Method	Semantic match
Bloomz	0%
keyBERT + PyDictionary	43%
YAKE + WordNet	28%
RAKE + WordNet	67%
keyBERT + WordNet	35%
keyBERT + Wiki	53%
RAKE+Wiki	67%
SimpleT5	70 %
YAKE+Wiki	66%
TextRank+Wiki	69%

Most of the methods performed really good on the smaller dataset, achieving a semantic match in most cases. Runs containing Simple T5 proved to be the best method, scoring around 70%. Wiki generally scored better than other methods, excluding SimpleT5, due to the large database it has the access to. All the methods showed the best results with single word terms.

Blooms variant Bloomz performed the worst with a score of 0, which can be explained with the fact that the Bloom run used only 100 examples due to token restrictions.

When the tests were repeated with the extended dataset, all of their results improved. The biggest improvement was with Bloomz model that achieved 71% on this test and 0% on the reduced dataset. Simple T5 remained as the best model with 73% semantic match. Runs and their semantic match percentages are shown in the table below.

Table 5: The table showing the percentage of correctly explained terms on extended dataset .

Extraction method	Semantic match
SimpleT5	73 %
keyBERT + Wiki	64%
RAKE+Wiki	68%
TextRank+Wiki	69%
Bloomz	71%
keyBERT + WordNet	44%
keyBERT + PyDictionary	51%
RAKE + WordNet	44%
YAKE+Wiki	69%
YAKE + WordNet	44%

However, since the number of evaluated terms differed significantly, it is also beneficial to take into the account the number of evaluated terms. Even though the semantic match percentage was lower for methods like TextRank + Wiki, the number of semantically matched definition was higher. In both datasets, smaller and larger, extended dataset methods using Word Net were consistently shown as the worst performers.

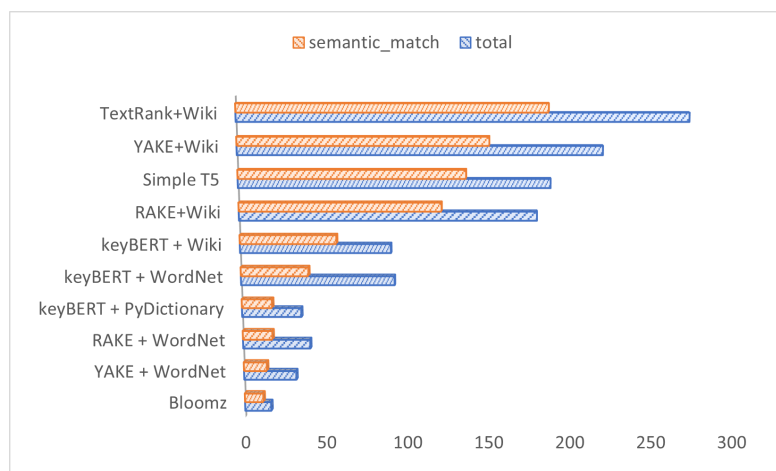
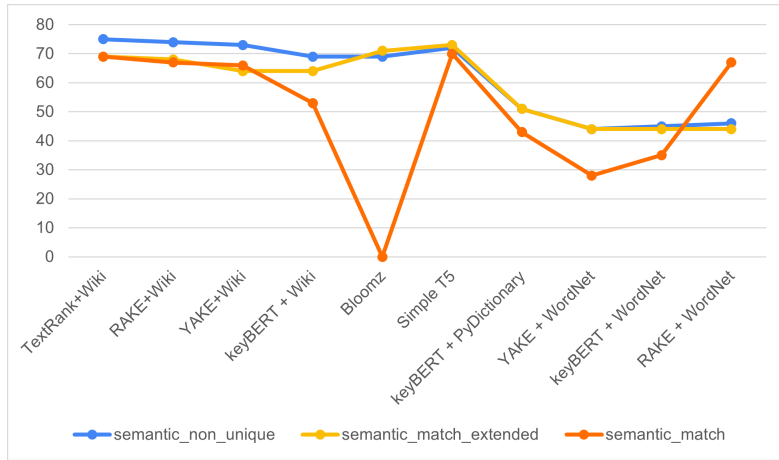


Figure 4: Relationship between the number of evaluated terms and the number of semantically matched definitions in the extended dataset.

The last dataset on which the examples were evaluated consisted of non-unique words that

appeared in the train dataset. Percentages for this dataset mostly followed the ones that were achieved on the extended dataset. With the largest deviation at around 5%. Methods are shown with their semantic matched for all three cases in the figure below.

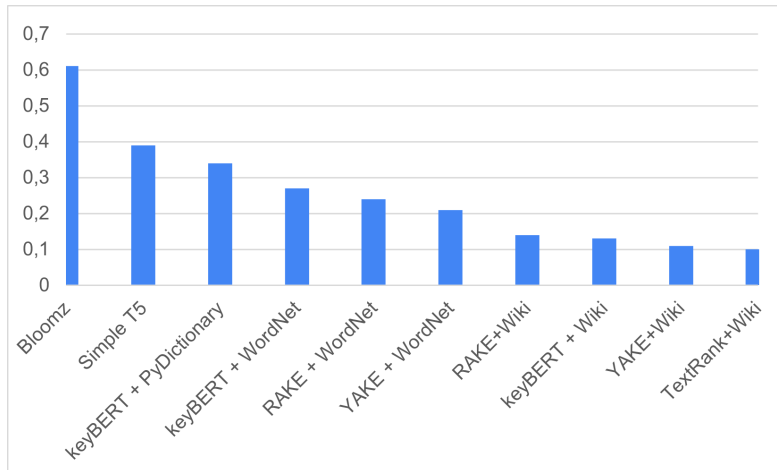


**Figure 5:** Percentages of correctly explained term in smaller dataset, extended test data set and non-unique words dataset.

Other than overall definition matching, models also assessed on their performance in abbreviation extension. In this aspect, Bloomz performed the best with a score of 61% semantic match. All runs using Wiki performed similarly, scoring around 10%. This was somewhat expected as abbreviations could have different meaning based on the context and Wikipedia can not decipher which one is correct. More powerful models as Bloomz are scoring better as they are trained to pay attention to the context of the term. The only model that had any exact or partial matches was Simple T5. It had 7 exact and 8 partial matches. All the results are shown in the table below.

Table 6: The table showing the percentage of correctly explained abbreviations.

Method	Semantic match
SimpleT5	39 %
keyBERT + Wiki	13%
RAKE+Wiki	14%
TextRank+Wiki	10%
Bloomz	61%
keyBERT + WordNet	27%
keyBERT + PyDictionary	34%
RAKE + WordNet	24%
YAKE+Wiki	11%
YAKE + WordNet	21%

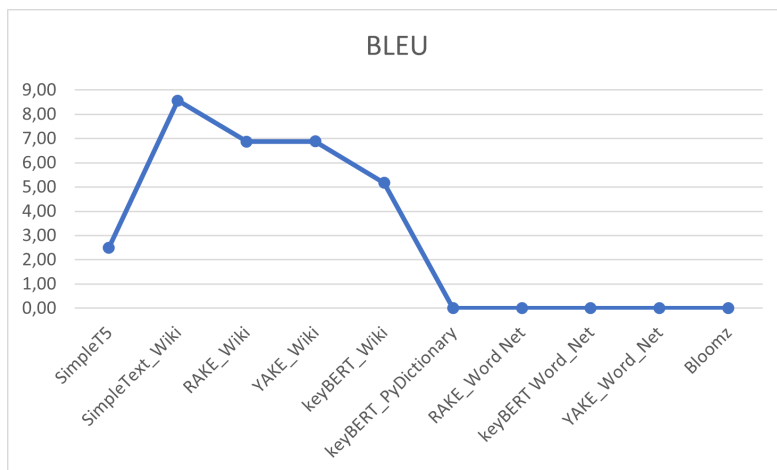


**Figure 6:** Percentages of correctly explained abbreviations.

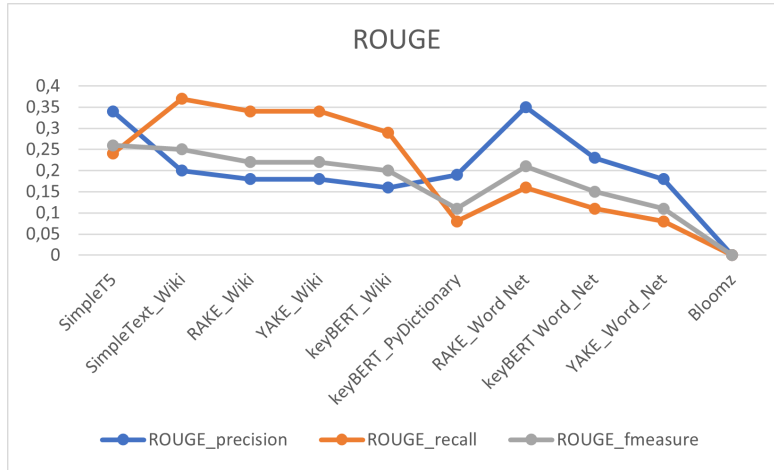
### 6.2.2. BLEU & ROUGE

BLEU is a score that is calculated by comparing the sentences to a number of reference sentences. The output of BLEU score is a numerical value between 0 and 1 which denotes how similar are the input sentences to the references. In this task, reference sentences represent the correct explanations for extracted difficult terms.

ROUGE score is a set of metric that compared the generated text to reference sentences. For this task 3 values contained in the set were used. Precision is a metric that calculates how close are the predicted definitions to the true definitions. Recall is the metric that calculates the proportion of the correct words used in predicted definitions. Fmeasure is a metric calculated when recall and precision are combined in one metric.



**Figure 7:** BLEU scores for explanations.



**Figure 8:** ROUGE scores for explanations.

Sentences mostly differed from the reference sentences, which is evident for both BLEU and ROUGE scores. On both scores the combination of SimpleT5 and Wiki scored the highest while the rest of the methods scored similarly low.

### 6.3. Sentence simplification

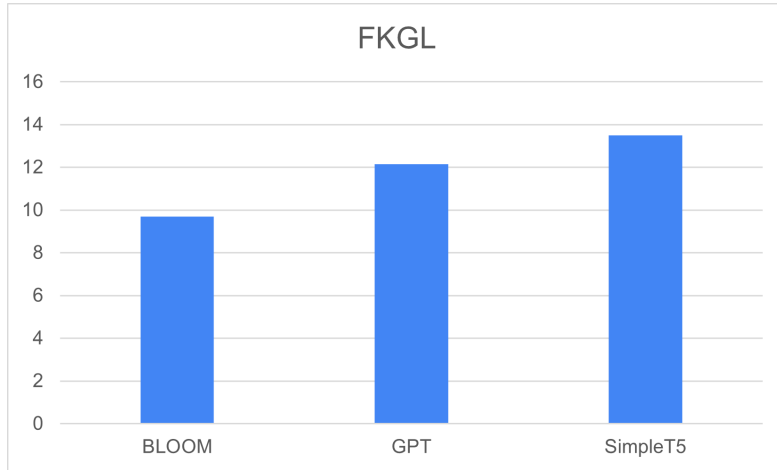
We only selected 100 sentences to simplify using GPT and Bloom due to token limitations. We simplified 648 sentences using Simple T5. For this task, the most metrics were used. The goal was to evaluate the results on their meaning and complexity, and to identify the instances in which the algorithm didn't change anything between the original and simplified sentence. The best method differed between each metric.

#### 6.3.1. FKGL

Flesch-Kincaid Grade Level (FKGL) is a metric used to measure the readability of generated text. It is used here to compare if the simplified sentences are more readable than the original. [3].

$$0.39 * \left( \frac{\text{total words}}{\text{total sentences}} \right) + 11.8 * \left( \frac{\text{total syllables}}{\text{total words}} - 15.59 \right) \quad (6)$$

In FKGL Simple T5 scored the best, with the score of 13, however the other models were really close with the scores at 12 and 9.



**Figure 9:** Models with respective FKGL scores

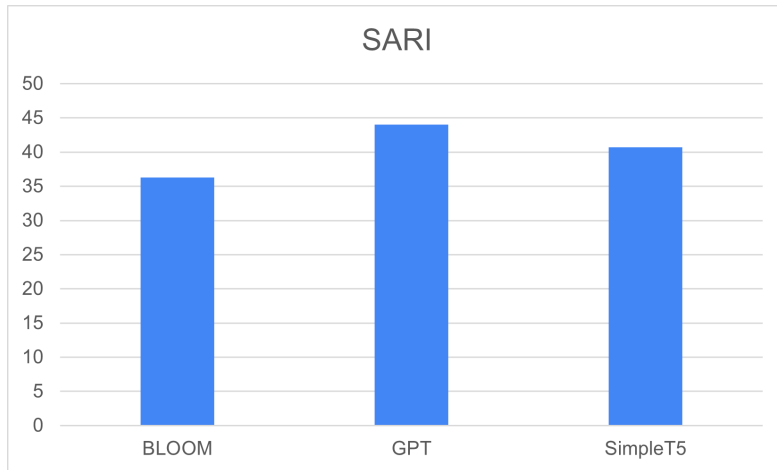
### 6.3.2. SARI

SARI[27] is a metric used for evaluating text simplification. The metric compares the predicted simplified sentences against the reference and the source sentences. It focuses on measuring the words that are added, deleted or kept.

$$Sari = \frac{(F1_{add} + F1_{keep}) + P_{del}}{3} \quad (7)$$

F1-add is f1 score for the words that were added F1-keep is the equivalent for the words that were kept P-del denotes the precision of deleted words

With SARI GPT scored the best with a score of 44, the second was Simple T5 with 40, and the last Bloom with 36.

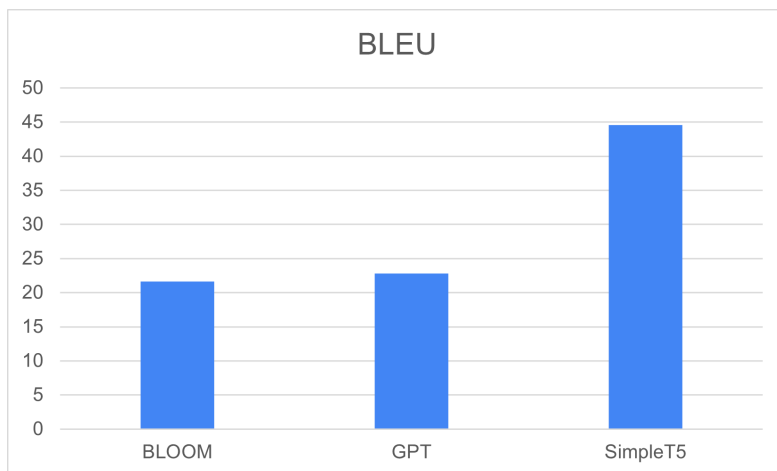


**Figure 10:** Models with respective SARI scores.

### 6.3.3. BLEU

BLEU is a score that is calculated by comparing the sentences to a number of reference sentences. The output of BLEU score is a numerical value between 0 and 1 which denotes how similar are the input sentences to the references. In this task, reference sentences represent the manually simplified sentences.

With BLEU Simple T5 did by far the best with a score of 44. The other modes scored 22 and 21.



**Figure 11:** Models with respective BLEU scores.

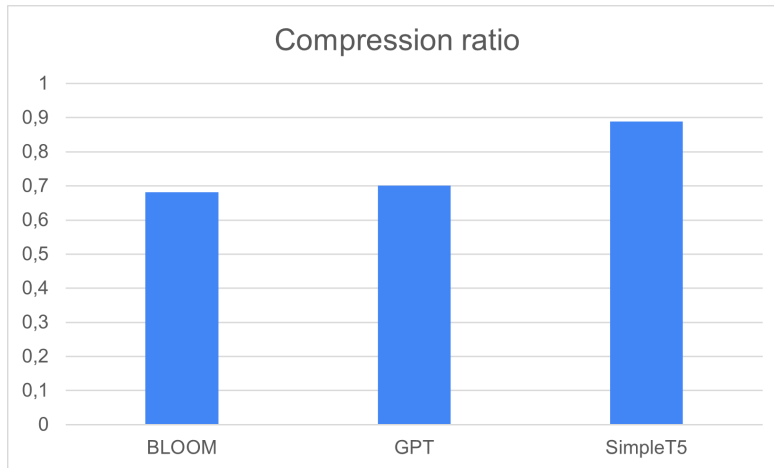


### 6.3.4. Compression ratio

Compression ratio is a metric that calculates how much a sentence was shortened.

$$compression = \frac{\text{Number Of Words Original}}{\text{Number Of Words Simplification}} \quad (8)$$

Simple T5 was also the model that compressed sentences the most with a compression rate of 89%, GPT also compressed sentences but with a much lower rate of 70%. Bloom did similarly with a compression rate of 68%.



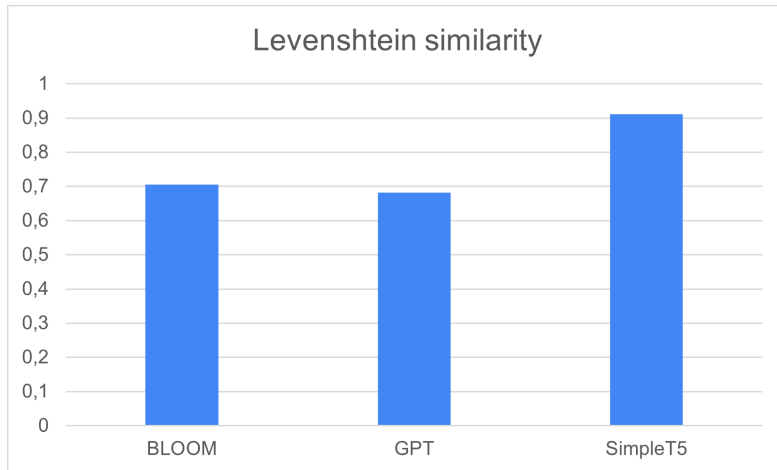
**Figure 12:** Models with respective compression rate.

All of the models had compression rate around 1 with only slight differences.

### 6.3.5. Levenshtein similarity

Levenshtein [28] similarity is a metric that calculates how many insertions or deletions would be needed to transform one text into another.

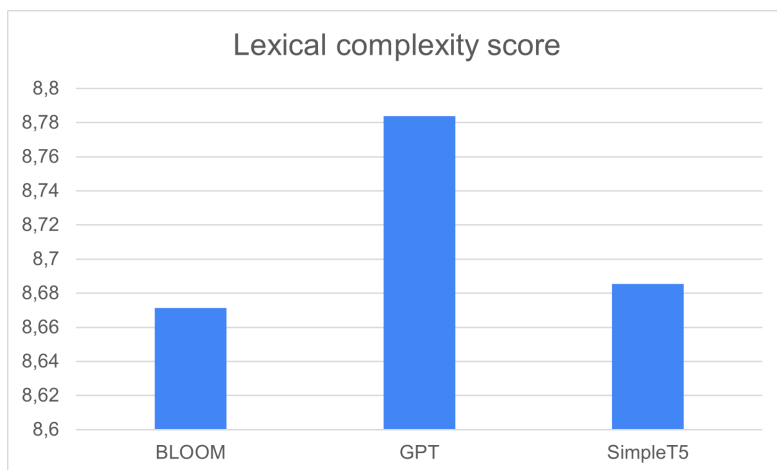
Levenshtein similarity was largest for Simple T5 where it was around 0,91. Second was Bloom with the score of 0,71 and the last was GPT with 0,68.



**Figure 13:** Models with respective Levenshtein similarity.

### 6.3.6. Lexical complexity score

Lexical complexity score reflect how complex the simplified text is to understand to the average person. This metric is also useful as a means of determining how much the text was simplified when compared to the original.



**Figure 14:** Models with respective lexical complexity.

The only model with any kind of exact match was simple T5 that had 1% match. All of the models has simillar lexical complexity at around 8,6.

## 7. Conclusion

In this report we have explained our approach for identifying, scoring and explaining difficult words. We have also shown 3 different approaches in simplifying whole sentences. We have submitted 10 runs for difficult term identification and scoring, 10 runs for term explanation and 3 for sentence simplification. We used a combination of models for each task, combining using prompts and training.

Furthermore, we compared the results obtained with all these methods and concluded that it is possible to achieve decent results with some of the methods, mainly Simple T5, which proved to be the best method overall. Still, there is room for improvement. Since our runs were not evaluated by us students, but by the teacher, we couldn't provide deeper understanding of some of the results. It would be beneficial for us to gain insight into how our results were evaluated and which terms were, for example, the most difficult for each of the methods to score. We think that this could be one of topic of future research.

Another topic that presented itself was, how appropriate was it to use automated metrics to score definitions. This is a topic that could be very interesting to explore since, in our research, semantic matches and automated metrics were scoring models very differently.

In the end, even though we only used free to use programs and our own equipment we achieved pretty good results, proving that NLP tasks are not only for GPT and can be attempted by anyone.

## References

- [1] L. Ermakova, E. SanJuan, S. Huet, O. Augereau, H. Azarbondyad, J. Kamps, Clef 2023 simpletext track, in: J. Kamps, L. Goeriot, F. Crestani, M. Maistro, H. Joho, B. Davis, C. Gurrin, U. Kruschwitz, A. Caputo (Eds.), *Advances in Information Retrieval*, Springer Nature Switzerland, Cham, 2023, pp. 536–545.
- [2] L. Ermakova, E. SanJuan, J. Kamps, S. Huet, I. Ovchinnikova, D. Nurbakova, S. Araújo, R. Hannachi, E. Mathurin, P. Bellot, Overview of the clef 2022 simpletext lab: Automatic simplification of scientific texts, in: A. Barrón-Cedeño, G. Da San Martino, M. Degli Esposti, F. Sebastiani, C. Macdonald, G. Pasi, A. Hanbury, M. Potthast, G. Faggioli, N. Ferro (Eds.), *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, Springer International Publishing, Cham, 2022, pp. 470–494.
- [3] J. Kincaid, R. Fishburne, R. Rogers, B. Chissom, Derivation of new readability formula for navy enlisted personnel, Millington, TN: Navy Research Branch (1975).
- [4] M.-S. Paukkeri, M. Ollikainen, T. Honkela, Assessing user-specific difficulty of documents, *Information Processing Management* 49 (2013) 198–212. URL: <https://www.sciencedirect.com/science/article/pii/S0306457312000532>. doi:<https://doi.org/10.1016/j.ipm.2012.04.001>.
- [5] J.-y. Kim, Predicting l2 writing proficiency using linguistic complexity measures: A corpus-based study., *English Teaching* 69 (2014).
- [6] Q. Li, H. Peng, J. Li, C. Xia, R. Yang, L. Sun, P. Yu, L. He, A survey on text classification: From

- traditional to deep learning, *ACM Transactions on Intelligent Systems and Technology* 13 (2022) 1–41. doi:10.1145/3495162.
- [7] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, *arXiv preprint arXiv:1810.04805* (2018).
- [8] R. Dale, Gpt-3: What’s it good for?, *Natural Language Engineering* 27 (2021) 113–118. doi:10.1017/S1351324920000601.
- [9] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, *Advances in neural information processing systems* 27 (2014).
- [10] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, *arXiv preprint arXiv:1409.0473* (2014).
- [11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, *Advances in neural information processing systems* 30 (2017).
- [12] M. Grootendorst, Keybert: Minimal keyword extraction with bert., 2020. URL: <https://doi.org/10.5281/zenodo.4461265>. doi:10.5281/zenodo.4461265.
- [13] S. Rose, D. Engel, N. Cramer, W. Cowley, Automatic Keyword Extraction from Individual Documents, 2010, pp. 1 – 20. doi:10.1002/9780470689646.ch1.
- [14] S. Bird, E. Klein, E. Loper, Natural language processing with Python: analyzing text with the natural language toolkit, ” O’Reilly Media, Inc.”, 2009.
- [15] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, A. Jatowt, Yake! keyword extraction from single documents using multiple local features, *Information Sciences* 509 (2020) 257–289. URL: <https://www.sciencedirect.com/science/article/pii/S0020025519308588>. doi:<https://doi.org/10.1016/j.ins.2019.09.013>.
- [16] BigScience Workshop, BLOOM (revision 4ab0472), 2022. URL: <https://huggingface.co/bigscience/bloom>. doi:10.57967/hf/0003.
- [17] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, *Journal of Machine Learning Research* 21 (2020) 1–67. URL: <http://jmlr.org/papers/v21/20-074.html>.
- [18] R. Mihalcea, P. Tarau, TextRank: Bringing order into text, in: *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Barcelona, Spain, 2004, pp. 404–411. URL: <https://aclanthology.org/W04-3252>.
- [19] F. Boudin, pke: an open source python-based keyphrase extraction toolkit, in: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, Osaka, Japan, 2016, pp. 69–73. URL: <http://aclweb.org/anthology/C16-2015>.
- [20] R. Flesch, How to write plain english, University of Canterbury. Available at [http://www.mang.canterbury.ac.nz/writing\\_guide/writing/flesch.shtml](http://www.mang.canterbury.ac.nz/writing_guide/writing/flesch.shtml). [Retrieved 5 February 2016] (1979).
- [21] M. Coleman, T. L. Liau, A computer readability formula designed for machine scoring., *Journal of Applied Psychology* 60 (1975) 283.
- [22] R. Senter, E. A. Smith, Automated readability index, Technical Report, Cincinnati Univ OH, 1967.
- [23] J. S. Chall, E. Dale, Readability revisited: The new Dale-Chall readability formula, Brookline

Books, 1995.

- [24] C. Fellbaum, WordNet: An Electronic Lexical Database, Bradford Books, 1998. URL: <https://mitpress.mit.edu/9780262561167/>.
  - [25] K. Papineni, S. Roukos, T. Ward, W.-J. Zhu, Bleu: A method for automatic evaluation of machine translation, in: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02, Association for Computational Linguistics, USA, 2002, p. 311–318. URL: <https://doi.org/10.3115/1073083.1073135>. doi:10.3115/1073083.1073135.
  - [26] C.-Y. Lin, ROUGE: A package for automatic evaluation of summaries, in: Text Summarization Branches Out, Association for Computational Linguistics, Barcelona, Spain, 2004, pp. 74–81. URL: <https://www.aclweb.org/anthology/W04-1013>.
  - [27] Optimizing statistical machine translation for text simplification, volume 4, 2016, pp. 401–415. URL: <https://www.aclweb.org/anthology/Q16-1029>.
  - [28] L. Yujian, L. Bo, A normalized levenshtein distance metric, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (2007) 1091–1095. doi:10.1109/TPAMI.2007.1078.
- Here you can find prompts that were used for Bloom and GPT.

## A. Difficult concept identification

### A.1. Bloom

Sentence: However, in information-centric networking (ICN) the end-to-end encryption makes the content caching ineffective since encrypted content stored in a cache is useless for any consumer except those who know the encryption key.\n\n Term: content caching.\n\n \n\n Sentence: Quantum circuits for arithmetic functions over Galois fields such as squaring are required to implement quantum cryptanalysis algorithms.\n\n.\n\n Term: quantum cryptanalysis.\n\n \n\n \n\n Sentence: The XECGA is then used to build the probabilistic model and to sample a new population based on the probabilistic model.\n\n Term: probabilistic model\n\n \n\n \n\n Sentence: We also present a subset demonstration of this model, TravelToken, which utilizes QR code that stores and uses travel information in smart contract over Ethereum.\n\n Term: Ethereum\n\n \n\n \n\n Sentence: Treating search engines as editorial products with intrinsic biases can help understand the structure of information flows in new media.\n\n Term: intrinsic\n\n \n\n \n\n Sentence: Penetration tests have become a valuable tool in the cyber security defence strategy in terms of detecting vulnerabilities.\n\n Term:

### A.2. GPT

”Find up to 5 difficult words:\n\n”+input

## B. Difficult concept explanation

### B.1. Bloom

Term: content caching \n\ Explanation: Content caching is a performance optimization mechanism in which data is delivered from the closest servers for optimal application performance.\n\ \n\

Term: logic qubit \n\ Explanation: A logical qubit is a physical or abstract qubit that performs as specified in a quantum algorithm or quantum circuit subject to unitary transformations, has a long enough coherence time to be usable by quantum logic gates.\n\ \n\

Term: quantum cryptanalysis \n\ Explanation: Quantum cryptanalysis is the study and evaluation of cryptographic algorithms in the presence of a quantum enabled adversary.\n\ \n\

Term: probabilistic model \n\ Explanation: Probabilistic modeling is a statistical technique used to take into account the impact of random events or actions in predicting the potential occurrence of future outcomes. \n\ \n\

Term: cyber-security \n\ Explanation: Ethereum is a decentralized, open-source blockchain with smart contract functionality. Ether is the native cryptocurrency of the platform.\n\ \n\

Term: intrinsic \n\ Explanation:

### B.2. GPT

"Explain the meaning of these words:\n\n"+input

## C. Sentence simplification

### C.1. Bloom

Sentence: In the modern era of automation and robotics, autonomous vehicles are currently the focus of academic and industrial research.\n\ Simplification: Current academic and industrial research is interested in autonomous vehicles.\n\ \n\

Sentence: With the ever increasing number of unmanned aerial vehicles getting involved in activities in the civilian and commercial domain, there is an increased need for autonomy in these systems too.\n\ Simplification: Drones are increasingly used in the civilian and commercial domain and need to be autonomous.\n\ \n\

Sentence: Due to guidelines set by the governments regarding the operation ceiling of civil drones, road-tracking based navigation is garnering interest.\n\ Simplification: Governments set guidelines on the operation ceiling of civil drones. So, road-tracking based navigation is attracting interest.\n\ \n\

Sentence: In an attempt to achieve the above mentioned tasks, we propose an imitation learning based, data-driven solution to UAV autonomy for navigating through city streets by learning to fly by imitating an expert pilot.\n\ Simplification: Researchers propose data-driven solutions allowing drones to autonomously navigate city streets, learning to fly by imitating an expert pilot.\n\ \n\

Sentence: Based on the Inception-v3 architecture, our system performs better in terms of processing complexity and accuracy than many existing models for imitation learning.\n\ Simplification: The Inception-v3 architecture has better accuracy than many existing models of imitation learning.\n\ \n\

Sentence: Permissions were taken from required authorities who made sure that minimal risk (to pedestrians) is involved in the data collection process.\n\ Simplification:

## C.2. GPT

"Simplify this sentence:\n\n"+input