University of Split and University of Malta (Team AB&DPV) at the CLEF 2024 SimpleText Track: Scientific Text Made Simpler Through the Use of Artificial Intelligence

Notebook for the SimpleText Lab at CLEF 2024[1] by Team AB&PDV

Antonia Bartulović[1*†] and Dóra Paula Váradi[2*†]

[1] *University of Split Ul. Ruđera Boškovića 31, 21000, Split, Croatia*
[2] *University of Malta, Msida MSD 2080, Malta*

Abstract

This paper describes the participation of the AB&DPV Team of the CLEF 2024 SimpleText track, which aims to simplify scientific texts. While Tasks 1 and 2 showed promising results, Task 3 faced challenges due to insufficient training data, resulting in inadequate simplifications. Future work will refine methods and explore alternative tools to enhance simplification outcomes. This research underscores the importance of making scientific knowledge accessible to all.

Keywords

CLEF, SimpleText, Natural Language Processing, Automatic Simplification

1. Introduction

1.1. Introduction and overview

The simplification of scientific texts is crucial for making scientific knowledge accessible to a broader audience, including non-experts and those with limited literacy skills. The CLEF 2024 SimpleText Track [1, 2, 3, 4] addresses this challenge by organizing three specific tasks aimed at improving access to scientific texts:

- Task 1: What is in (or out)? Selecting passages to include in a simplified summary [5].
  - This Lexical Simplification focuses on replacing complex words and phrases with simpler alternatives without changing the overall meaning of the text.
- Task 2: What is unclear? Difficult concept identification and explanation (definitions, abbreviation deciphering, context, applications...) [6].
  - This Syntactic Simplification aims to restructure sentences to make them easier to read and understand, often by breaking down complex sentences into simpler, shorter ones.
- Task 3: Rewrite this! Given a query, simplify passages from scientific abstracts [7].
  - This Full Text Simplification integrates both lexical and syntactic simplification to produce comprehensively simplified versions of scientific texts.

The research focuses on all three tasks, which together present a comprehensive challenge in simplifying scientific texts. By addressing these tasks, we aim to improve both the vocabulary and the structure of sentences, enhancing readability while preserving the integrity of the scientific information. This paper details the approach taken, which employs advanced natural language processing techniques to tackle these tasks. The current state-of-the-art [8] is also discussed in text simplification, referencing key studies.

[1] *CLEF 2024: Conference and Labs of the Evaluation Forum, September 09–12, 2024, Grenoble, France*
* Corresponding author.
† These authors contributed equally.
antonia.bartulovic.00@fesb.hr (A. Bartulović); dora.varadi.21@um.edu.mt (D. P. Váradi)

This research is motivated by the goals of enabling all to access information, as it is a fundamental right for all. Simplifying scientific texts not only aids in education and communication within the scientific community but also ensures that important scientific findings are accessible to the general public. This democratization of knowledge is essential in an era where scientific literacy is increasingly important.

1.2 State-of-the-Art Overview

The field of text simplification has seen significant advancements in recent years, particularly with the advent of transformer-based models and deep learning techniques. Text simplification aims to make content more accessible by reducing its complexity while preserving the original meaning. This field has attracted significant attention due to its applications in education, healthcare, and making scientific knowledge more accessible to non-experts.

1.2.1. Neural Network Approaches

Recent methods leverage deep neural networks for automatic text simplification. For instance, the transformer model, introduced by Vaswani et al. in 2017, has been adapted for text simplification tasks. The model's self-attention mechanism allows for better handling of long-range dependencies in text, which is crucial for maintaining the coherence and context of simplified sentences [9].

1.2.2. Pre-trained Language Models

Large pre-trained language models like BERT by Devlin et al. [10] and GPT-3 by Brown et al. [11] have demonstrated significant improvements in natural language understanding tasks, including text simplification. These models are fine-tuned on simplification datasets to enhance their performance, as seen in the work by Zhang and Lapata in 2017 [12], where they utilized a sequence-to-sequence architecture to simplify texts effectively.

1.2.3. Lexical and Syntactic Simplification

Traditional approaches to text simplification focus on lexical and syntactic transformations. Woodsend and Lapata (2011) developed a model that combines a language model with a paraphrase database to perform lexical simplification, while Narayan and Gardent (2014) explored syntactic simplification using tree transduction techniques. These methods aim to replace complex words and restructure sentences to enhance readability [13, 14].

1.2.4. Evaluation Metrics

Evaluating text simplification systems remains a challenge due to the subjective nature of readability and simplicity. Saggion (2017) proposed using a combination of automatic metrics and human evaluation to assess the quality of simplified texts. Common metrics include SARI (System output Against References and against the Input sentence), which measures the goodness of words added, deleted, and kept by the simplification system [15].

2. Approach

2.1. Data Description

The data used in the research for the CLEF 2024 SimpleText track [4, 1] consists of scientific texts from various domains, provided by the task organizers. The dataset includes documents that require simplification at different levels: lexical, syntactic, and full text. The specifics of the data used for each task are outlined below, referencing the guidelines provided by the CLEF SimpleText track organizers.

Task 1:

- Source: The dataset for lexical simplification includes scientific texts with complex words identified by domain experts.
- Annotations: Each complex word is annotated with simpler alternatives, assisting in training models to recognize and replace difficult words while maintaining context.
- Format: Data is provided in a tab-separated format with columns for the original sentence, the complex word, and the list of simpler alternatives.

Task 2:

- Source: The dataset for syntactic simplification includes sentences from scientific texts that are structurally complex.
- Annotations: Each sentence is paired with a syntactically simplified version, created by professional annotators to ensure the simplified sentences retain the original meaning.
- Format: Data is provided in a CSV format with columns for the original sentence and the simplified sentence.

Task 3:

- Source: The dataset for full text simplification comprises entire paragraphs or sections from scientific articles that require both lexical and syntactic simplification.
- Annotations: Each paragraph is accompanied by a simplified version, annotated to reflect both lexical and syntactic changes, ensuring that the simplified text is easier to read and understand while preserving the scientific content.
- Format: Data is provided in a JSON format, with fields for the original text and the simplified text.

2.2. Methodology

The code was ran locally using Python version 3.8.19 due to errors building certain packages with higher versions.

2.2.1. Data Preprocessing

Several preprocessing steps were performed to prepare the data for the models:

Tokenization: Texts were tokenized into sentences and words using standard natural language processing (NLP) libraries.

Normalization: All texts were normalized to lower case, and punctuation was standardized to ensure consistency across the dataset.

Filtering: Sentences or paragraphs that did not meet certain quality criteria (e.g., very short or very long sentences) were filtered out to maintain a manageable and high-quality dataset.

2.2.2 Training and Validation Split

The dataset was split into training and validation sets to evaluate the performance of the models. An 80-20 split was used [16], with 80% of the data used for training and 20% reserved for validation. This split ensures that the models are tested on a diverse set of examples, providing a robust evaluation of their simplification capabilities.

2.2.3. Task 1

The approach taken for Task 1, which involves lexical simplification, uses a combination of information retrieval and readability assessment techniques to identify and simplify complex passages from scientific texts. Here is an expanded explanation of the approach:

Query Execution: Predefined queries are executed to retrieve relevant scientific abstracts. For each query, an HTTP GET request is made to a search API [17] endpoint to retrieve documents from the database, with a maximum of 100 documents requested per query.

Retrieval and Scoring: The response from the API is parsed to extract the list of document hits. The scores of the retrieved documents are normalized by calculating the minimum and maximum scores and then applying a min-max normalization to each document score.

Readability Assessment: For each document, the Flesch-Kincaid Grade Level (FKGL) [18] of the abstract is calculated to estimate the grade level required to understand the text. A dictionary is created for each passage, containing various attributes such as `run_id`, `topic_id`, `query_id`, `doc_id`, `rel_score` (normalized relevance score), and the `passage` itself (first 1000 characters of the abstract).

Readability Reranking: The passages, along with their normalized document scores and readability scores, are stored in a list. Each passage entry includes information such as the query ID, document ID, normalized relevance score, and the extracted passage content.

Readability Normalization: The FKGL scores are normalized similarly to the document scores, and each passage's FKGL score is normalized accordingly. The FKGL scores are also normalized using min-max normalization. This step ensures that the readability scores are on the same scale as the document scores, facilitating a fair comparison and combination of these metrics.

Combination Score: The normalized FKGL score is assigned as the combination score (`comb_score`) for each passage. This score is intended for further processing, such as ranking passages based on their readability.

The results were saved into a JSON file. This methodology ensures that the retrieved passages are not only relevant to the query but also assessed for readability, facilitating the task of lexical simplification. Before the results are submitted, the run ID for each result is updated to reflect the specific task and methodology used. This step ensures that the results are properly labelled and can be easily identified during evaluation

The approach for Task 1 involves a systematic process of document retrieval, readability assessment, and reranking based on the FKGL scores. By normalizing both the document relevance scores and readability scores, the method ensures that the passages are ranked effectively. The final output consists of the most readable passages, prioritized for their simplicity and accessibility. This method ensures that complex scientific texts are made easier to read, thereby increasing their accessibility to a broader audience.

2.2.4. Task 2

Task 2 is divided into two subtasks:

- Subtask 1: Extracting and merging relevant terms, definitions, and explanations.
- Subtask 2: Preprocessing text, computing BLEU scores for similarity, and selecting the best definitions and explanations.

Task 2 Subtask 1: Extracting and Merging Relevant Data

Data Loading: The necessary datasets are loaded into DataFrames using `pandas` [19], including definitions, explanations, generated definitions, documents, and terms.

Data Merging: The `documents` and `terms` DataFrames are merged to associate terms with their respective documents. Unnecessary columns are dropped, and new columns are added to standardize the data format.

Task 2 Subtask 2: Text Preprocessing, BLEU Score Computation, and Best Definition Selection

Text Preprocessing: The `preprocess` function standardizes using TensorFlow [20] the text by converting it to lowercase, removing punctuation, tokenizing, removing stop words, and stemming the tokens.

BLEU Score Computation [21]: The `compute_bleu` function calculates the BLEU score between a reference and a candidate definition to measure their similarity. The BLEU score is used to compare different definitions and explanations for the same term.

Best Definition Selection: The `select_best_definition` function selects the best definition or explanation from a list by computing BLEU scores between all pairs and choosing the one with the highest score.

Term Matching: The `find_matching_term` function finds the best matching term in the dataset using fuzzy matching [22] if the exact term is not found.

Integrating Results: The terms, definitions, and explanations are grouped, and the best definition and explanation for each term are selected using the `select_best_definition` function. If multiple definitions or explanations are present, the one with the highest BLEU score is chosen. If the BLEU scores are identical, the shorter definition is selected.

Assigning Definitions and Explanations: A function is developed to find the best matching term from the definitions and explanations dataset. This ensures that even if a term is not an exact match, the closest possible term is used. The processed data is used to update the term explanations data frame with the best definitions and explanations. This involves iterating through each term and applying the best match based on the BLEU score and other criteria.

Creating the Final Output: The results are formatted into a list of dictionaries, each containing the run ID, manual flag, sentence ID, term, difficulty, and the selected definition and explanation (if applicable). This structured format is suitable for submission.

The approach for Task 2 leverages both manually curated and automatically generated data to provide comprehensive definitions and explanations for scientific terms. By using BLEU scores to select the best definitions, the method ensures that the explanations are not only accurate but also succinct. This process enhances the readability and understandability of scientific texts, making them more accessible to a broader audience.

2.2.5. Task 3

Task 3 focuses on assessing and simplifying both sentence-level and document-level texts using deep learning models. This involves training models to simplify sentences and abstracts, followed by evaluating the readability and quality of the simplified texts.

Task 3 Subtask 1: Sentence-level Simplification

Data Preparation: Load and merge the source and reference sentences from the training dataset. Tokenizers are created to transform text data into numerical sequences. One tokenizer is used for

the source (complex sentences) and another for the target (simplified sentences). The tokenizers are fitted on the respective text data to build a vocabulary and convert the text into sequences of integers. The size of the vocabularies for both source and target texts is determined. Since the lengths of sentences can vary, the sequences are padded to a fixed length to ensure uniformity in the input data for the model.

Model Architecture: A neural network model is defined, consisting of embedding layers, Bidirectional Long Short-Term Memory (LSTM) layers [23] to capture both forward and backward dependencies in the text. The model includes dropout layers to prevent overfitting and a TimeDistributed dense layer for the final output. The model is compiled with an appropriate loss function and optimizer, then trained on the prepared data. Training involves multiple epochs where the model learns to map complex sentences to their simplified versions. The model is compiled with the Adam optimizer and trained using the sparse categorical cross-entropy loss function.

Training: The model is trained for 20 epochs with a batch size of 32 and a validation split of 10%.

Testing: The trained model is used to predict the simplified sentences for the test dataset. Predictions are decoded to obtain the simplified sentences in text form.

Evaluation: The readability of the simplified sentences is evaluated using BLEU scores and other readability metrics.

Task 3 subtask 2: Document-level Simplification

Data Preparation: Load and merge the source and reference abstracts from the training dataset. Tokenize the abstracts to convert them into sequences of integers and pad the sequences to ensure uniform input length for the model.

Model Architecture: A sequential model with LSTM layers and an embedding layer is used for document-level simplification. The model is trained similarly to the sentence-level model but with longer input sequences.

Training and Testing: The model is trained and tested in a manner similar to Subtask 3.1, but with document-level inputs and outputs.

3. **Results**

3.1. Task 1

Given that the combined score calculation was based on the Flesch-Kincaid Grade Level, a metric designed for measuring readability, the results are somewhat relevant. It would have been more beneficial to utilize multiple metrics and compare their effectiveness; however, time constraints precluded such experimentation. FKGL, which considers word length and sentence length, was a logical choice for the task and provided sufficient results. The "score" field is a result of ElasticSearch's own calculations, which also serve as an effective scoring system for query relevancy.

Table 1: Relevance Results Evaluated on Test qrels

runid	MRR	Precision 10	Precision 20	NDCG 10	NDCG 20	Bpref	MAP
AB_DPV_SimpleText_task1_results_FKGL	0,6173	0,3733	0,2900	0,2818	0,2442	0,1966	0,1078

Table 2: Average Scores

Average rel_score	Average comb_score
0.23133980949662628	0.3396115556507078

3.2. Task 2

Task 2 was divided into three subtasks. Subtask 2.3 was not completed due to a lack of data. Subtasks 2.1 and 2.2, however, proved to be interesting in their own right. The solution was not implemented on test data. Proper implementation would involve using natural language processing to extract difficult terms from passages, followed by generating definitions for them or retrieving them from sources such as Wikipedia. As it stands, the definition extraction was conducted solely through fuzzy searching difficult terms from premade sources, yielding a satisfactory result, although it was not fully implemented on the test data and some definitions are not perfect.


```
{
    "run_id": "AB&DPV_Task2.2_BLEU",
    "manual": 0,
    "snt_id": "G01.1_1533716782_2",
    "term": "laptop",
    "difficulty": "e"
},
```


Figure 1: Example of difficulty “e” (easy)


```
{
    "run_id": "AB&DPV_Task2.2_BLEU",
    "manual": 0,
    "snt_id": "G10.1_2093013061_3",
    "term": "compute",
    "difficulty": "m"
},
```


Figure 2: Example of difficulty “m” (medium)


```
{
    "run_id": "AB&DPV_Task2.2_BLEU",
    "manual": 0,
    "snt_id": "G08.1_2889349357_1",
    "term": "JavaScript",
    "difficulty": "d",
    "definition": "A scripting or programming language that allows to implement
complex features on web page.",
    "explanation": "This term specifically designates a  programming language
which is useful for developers and which is mostly used by them."
},
```


Figure 3: Example of difficulty “d” (difficult) with Definition and Explanation

```
    {
        "run_id": "AB&DPV_Task2.2_BLEU",
        "manual": 0,
        "snt_id": "G04.1_2339209416_2",
        "term": "algorithms",
        "difficulty": "d",
        "definition": "A set
of\u00a0mathematical\u00a0instructions\u00a0or\u00a0rules\u00a0that,\u00a0especially
\u00a0if given to a\u00a0computer,
will\u00a0help\u00a0to\u00a0calculate\u00a0an\u00a0answer\u00a0to a\u00a0problem.",
        "explanation": null
    },
```


Figure 4: Example of difficulty "d" (difficult) with Faulty Definition and Explanation

3.3. Task 3

The goal of this task was to simplify sentences and documents. The chosen method yielded inadequate and often illegible simplifications. Training a model to predict simplified sentences based on the input using LSTM layers proved insufficient, as the provided training data was inadequate for such an approach. The model was unable to discern underlying meaning between words, despite ample training time, leading to output consisting of seemingly random words. A more effective approach might have been to investigate other text simplification tools, such as utilizing large language models (LLMs) like LLAMA or employing an entirely different implementation.

The training process of the model for the document-level simplification task yielded significant insights into its performance. The train performance scores, which provide a more comprehensive understanding of the model's learning behaviour and potential areas for improvement.

Table 3: Performance scores

Epoch	Time (s)	s/step	Training Loss	Validation Loss
1/20	18	3	8.0658	7.8402
2/20	19	4	7.1849	6.1243
3/20	19	4	5.5109	4.5786
4/20	20	4	3.9234	3.0568
5/20	19	4	2.5129	1.9095
6/20	20	4	1.5586	1.2988
7/20	20	4	1.1272	1.1248
8/20	20	4	1.0204	1.1042
9/20	20	4	1.0073	1.1072
10/20	15	3	1.0043	1.106
11/20	15	3	0.9976	1.1002
12/20	16	3	0.9898	1.0941
13/20	14	3	0.9837	1.0908
14/20	15	3	0.9813	1.0891
15/20	15	3	0.9795	1.087
16/20	16	3	0.9772	1.085
17/20	16	3	0.9749	1.0838
18/20	17	3	0.9719	1.0821
19/20	16	3	0.969	1.0796

20/20	15	3	0.9668	1.077

Overall, the training and validation losses showed a consistent decline, demonstrating the model's capability to learn and generalize well. The slight gap between the training and validation losses suggests some level of overfitting, which might be addressed with techniques such as dropout or regularization.


```
{
    "run_id": "AB&DVP_Task3.1_SequentialLSTM",
    "manual": 0,
    "snt_id": "G11.1_2892036907_5",
    "simplified_snt": "desire desire mec mec enzymes enzymes enzymes enzymes"
},
{
    "run_id": "AB&DVP_Task3.1_SequentialLSTM",
    "manual": 0,
    "snt_id": "G11.1_2892036907_6",
    "simplified_snt": "desire desire mec mec mec enzymes enzymes parties parties"
},
```


Figure 5: Example of Faulty Output of Simplification Output

Section 4. Conclusions

This working paper presented the approach and findings from the CLEF 2024 SimpleText Track, focusing on three tasks aimed at simplifying scientific texts. Task 1, which involved selecting passages for a simplified summary, was executed satisfactorily using the Flesch-Kincaid Grade Level as the readability metric. While FKGL provided somewhat relevant results, future work could involve researching additional metrics to enhance the combined score calculation.

Task 2, which aimed to identify and explain difficult concepts, was partially completed. Subtasks 2.1 and 2.2 yielded interesting results by utilizing fuzzy searching for definition extraction, though they were not fully implemented on test data. Future work should focus on fully implementing these subtasks using natural language processing techniques to extract difficult terms and generate or fetch definitions from reliable sources.

Task 3, which sought to simplify sentences and documents, was the most challenging and yielded inadequate results. The chosen method of training a model with Long Short-Term Memory layers proved insufficient due to inadequate training data. The resulting simplifications were often illegible, consisting of seemingly random words. Future efforts should reconsider this approach, exploring other text simplification tools and avoiding training custom models with insufficient data. Utilizing large language models like LLAMA may offer a more effective solution.

Overall, this research underscores the importance of selecting appropriate methods and metrics for text simplification tasks. Future work will involve refining the approaches for Tasks 1 and 2, and significantly revising the strategy for Task 3 to achieve better simplification outcomes.

References

[1] L. Ermakova, T. Miller, A.-G. Bosser, V. M. Palma-Preciado, G. Sidorov and A. Jatowt, "Overview of CLEF 2024 SimpleText Track on Improving Access to Scientific Texts," in *Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Fifteenth International Conference of the CLEF Association (CLEF 2024)*, 2024.

[2] L. Ermakova, E. SanJuan, S. Huet, O. Augereau, H. Azarbonyad and J. Kamps, "CLEF 2023 SimpleText Track: What Happens if General Users Search Scientific Texts?," in *dvances in Information Retrieval: 45th European Conference on Information Retrieval, ECIR 2023, Dublin, Ireland, April 2–6, 2023, Proceedings, Part III*, 2023.

[3] L. Ermakova, E. SanJuan, S. Huet, H. Azarbonyad, G. M. Di Nunzio, F. Vezzani, J. D'Souza, S. Kabongo, H. Babaei Giglou, Y. Zhang, S. Auer and J. Kamps, "CLEF 2024 SimpleText Track: Improving Access to Scientific Texts for Everyone," in *Advances in Information Retrieval: 46th European Conference on Information Retrieval, ECIR 2024, Glasgow, UK, March 24–28, 2024, Proceedings, Part VI*, 2024.

[4] L. Ermakova, E. SanJuan, S. Huet, O. Augereau, H. Azarbonyad and J. Kamps, "Overview of SimpleText - CLEF-2023 track on Automatic Simplification of Scientific Texts," in *Proceedings of the Fourteenth International Conference of the CLEF Association (CLEF 2023)*, 2023.

[5] E. SanJuan and e. al., "Overview of the CLEF 2024 SimpleText Task 1: Retrieve passages to include in a simplified summary," in *Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2024)*, 2024.

[6] G. M. Di Nunzio and e. al., "Overview of the CLEF 2024 SimpleText Task 2: Identify and explain difficult concepts," in *Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2024)*, 2024.

[7] L. Ermakova and e. al., "Overview of the CLEF 2024 SimpleText Task 3: Simplify scientific text," in *Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2024)*, 2024.

[8] J. D'Souza and e. al., "Overview of the CLEF 2024 SimpleText Task 4: Track the state-of-the-art in scholarly publications," in *Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2024)*, 2024.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems,* pp. 5998-6008, 2017.

[10] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019.

[11] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell and e. al., "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165,* 2020.

[12] X. Zhang and M. Lapata, "Sentence simplification with deep reinforcement learning," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017.

[13] K. Woodsend and M. Lapata, "Learning to simplify sentences with quasi-synchronous grammar and integer linear programming," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 2011.

[14] S. Narayan, C. S. Gardent and C. Gardent, "Hybrid simplification using deep semantics and machine translation," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2014.

[15] H. Saggion, "Automatic text simplification," *Synthesis Lectures on Human Language Technologies,* vol. 10, no. 1, pp. 1-137, 2017.

[16] "Split Data Training and Testing Set," [Online]. Available: https://www.askpython.com/python/examples/split-data-training-and-testing-set.

[17] "API - Textstat," [Online]. Available: https://github.com/textstat/textstat .

[18] "Flesch-Kincaid Grade Level," [Online]. Available: https://en.wikipedia.org/wiki/Flesch%E2%80%93Kincaid_readability_tests.

[19] "Pandas," [Online]. Available: https://pandas.pydata.org/ .

[20] "TensorFlow," [Online]. Available: https://www.tensorflow.org/ .

[21] "BLEU," [Online]. Available: https://en.wikipedia.org/wiki/BLEU.

[22] "The fuzz," [Online]. Available: https://github.com/seatgeek/thefuzz .

[23] "Complete Guide to RNN, LSTM, and Bidirectional LSTM," [Online]. Available: https://dagshub.com/blog/rnn-lstm-bidirectional-lstm/.