Document Simplification

Team Sharingans at SimpleText: Fine-Tuned LLM based approach to Scientific Text Simplification

Objective:

This paper is about making complicated science texts easier to understand for people who aren't experts. The main goals are:

1. Finding and picking out the important parts of the text that need to be simplified.
2. Spotting difficult scientific words, figuring out how hard they are, and explaining them in simpler terms.
3. Making scientific summaries and sentences easier to read without changing their meaning.

Data Sources:

1. The study uses a collection of research paper summaries (DBLP abstracts), search query text, and ratings on how relevant or useful the information is.
2. It also includes external data from news sites like *The Guardian* and *Tech Xplore* for variety.

Training and Testing Data:

1. There are datasets that help train and test the system, which include sentences from science texts, important keywords, definitions, and simple explanations.
2. A test dataset has more than 500 examples where keywords need to be identified and explained.

Simplification Data:

1. There’s a set of paired data: original science sentences and abstracts alongside their simpler versions, to teach the system how to simplify the language.
2. There are separate datasets specifically for simplifying individual sentences and full abstracts.

Model Design and Techniques Used:

1. Finding Relevant Information:

a. *Using MS-Marco + GPT-3.5 for better results*:

I] A database finds the most relevant research summaries.

Ii] GPT-3.5 is then used to choose the best parts of those summaries for simplification.

b.Keyword Extraction with RAKE + ColBERT + GPT-3.5:

I] RAKE helps pick out important words from the text.

Ii] ColBERT improves how the documents are ranked.

Iii] GPT-3.5 is used to select the key passages for further work.

2. Understanding and Explaining Terms:

a.Fine-Tuned GPT-3.5 Turbo:

I] GPT-3.5 identifies important terms, decides how difficult they are, and provides simple explanations.

Ii] It's trained to improve its accuracy and work better on new examples.

b.*KeyBERT + Random Forest + Mistral-7B*:

i]KeyBERT helps find key terms.

ii]Random Forest is used to figure out how difficult those terms are.

iii]Mistral-7B is then used to generate simple explanations for those terms.

3.Simplifying Sentences:

a.Fine-Tuned GPT-3.5 Turbo:

i]GPT-3.5 simplifies sentences and abstracts, with separate training for each task.

b.BART Sequence-to-Sequence:

i]BART, a model from Meta, was tested for simplification but didn't perform as well.

c.PEGASUS Model:

i]PEGASUS, another model designed for summarizing, was better than BART but still not as good as GPT-3.5.

Experiments and Evaluation:

1. Finding the Right Information:

i]How it was measured: They used metrics like *Mean Reciprocal Rank (MRR)*, *NDCG*, *Precision*, and *MAP* to see how well the system was working.

ii]Results: There were some problems with precision because some data wasn't organized properly during training, which led to less accurate results.

2. Explaining Terms:

i]How it was measured: Metrics like *Recall*, *Precision*, and *BLEU* were used to check how well the system found and explained terms.

ii]Results: GPT-3.5 did a good job at finding relevant terms and giving good explanations, but it had some trouble when it came to longer definitions (measured by BLEU scores).

3. Simplifying Sentences:

i]How it was measured:They used metrics like *FKGL* (a readability measure), *SARI*, *BLEU*, *Compression Ratio* (how much the text was shortened), and *Lexical Complexity* to see how well the system simplified text.

ii]Results: GPT-3.5 was the best at simplifying both sentences and abstracts when compared to other models like BART and PEGASUS.

Key Findings:

1.GPT-3.5 Turbo's Performance:

Best results across tasks, especially for keyword extraction (Task 2) and text simplification (Task 3).

2.Limitations of Baselines:

BART and PEGASUS models struggled with simplification precision and coherence compared to GPT-3.5.

3.Task 1 Challenges:

Low precision scores indicate issues with passage selection using the MS-Marco-GPT pipeline.

4.Room for Improvement:

Simplified texts could still benefit from improved readability without information loss.

Challenges and Insights:

1. Dataset Limitations:

The dataset was small and required a lot of manual work to organize, which made it harder to get the best results during training.

2. Model Constraints:

The tools used (like the Hugging Face API) had some limits, which made it harder to fully explore other methods that could've worked better.

3. Optimization Challenges:

Using a small batch size and a high learning rate helped improve results, but it also needed a lot of tweaking to make sure the model didn't learn too much from the data and become overly specific (a problem called overfitting).

Future Possible Solutions:

1.Improved Fine-Tuning Pipelines:

Addressing manual curation issues to enhance Task 1 performance.

2.Hybrid Approaches:

Combining GPT-3.5 with additional ranking models like ColBERT for passage selection.

3.Dataset Expansion:

Using larger, high-quality datasets for better generalization and performance.

4.Automation and Scalability:

Automating keyword extraction pipelines to reduce dependency on APIs with limited requests.

Conclusion:

The study demonstrates that fine-tuned GPT-3.5 Turbo is highly effective for scientific text simplification tasks. However, challenges like dataset limitations and low precision in passage selection need to be addressed. Future work should focus on improving the generalization capabilities and automation of the pipelines to make scientific texts more accessible to diverse audiences.

Progressive Document-level Text Simplification via Large Language Models

Introduction

The field of text simplification aims to make text *more accessible for wider audiences*, including non-native speakers and individuals with cognitive impairments. While traditional simplification methods focus on lexical and sentence-level modifications, long document-level text simplification (DS) remains underexplored. Large Language Models (LLMs), such as ChatGPT, have demonstrated potential in natural language processing tasks but struggle with DS, often treating it as document summarization. The study proposes a Progressive Simplification Method (ProgDS) to address this limitation, inspired by human editing strategies. ProgDS hierarchically simplifies documents at discourse, topic, and lexical levels.

Objective-

1. **To address the challenges of long document simplification, such as maintaining content preservation and resolving ambiguity.**
2. **To develop the ProgDS framework, which performs stepwise simplifications from the discourse to the lexical level.**
3. **To compare ProgDS against existing simplification methods on benchmarks like Wiki-auto and Newsela datasets and evaluate its effectiveness in simplifying long documents while maintaining coherence and faithfulness.**

Model Design

The ProgDS Framework involves three hierarchical levels:

1. **Discourse-Level Simplification: Divides documents into multiple topics with subheadings, enhancing structural organization and global coherence.**
2. **Topic-Level Simplification: Simplifies paragraphs while ensuring logical flow within and between sentences.**
3. **Lexical-Level Simplification: Focuses on simplifying complex vocabulary and expressions to improve readability.**

Additionally, ProgDS integrates iterative simplification to refine results progressively. Techniques like in-context learning (ICL) and chain-of-thought (COT) prompting enhance LLMs' ability to follow human-like reasoning in simplification tasks.

Dataset Development

1. **Datasets Used:**
   - **Wiki-auto: Contains automatically aligned complex-simple document pairs from Wikipedia and Simple English Wikipedia.(Link-:** Wiki auto)
   - **Newsela-auto: A high-quality, human-annotated dataset with multiple simplification levels for news articles.**
2. **Preprocessing:**
   - **Long documents (over 1,000 tokens) were extracted from Newsela-auto to create a focused evaluation dataset.**
   - **Paragraphs in Wiki-auto were realigned to match the hierarchical framework of ProgDS.**

Challenges and Insights

1. **Content Preservation: LLMs face difficulties in retaining essential information while simplifying long texts, often producing overly short outputs resembling summaries.**
2. **Ambiguity: Simplification involves subjective decisions, making it challenging for LLMs to consistently meet human expectations.**
3. **Dataset Quality: Proprietary datasets like Newsela-auto outperform open datasets like Wiki-auto in terms of alignment and quality, highlighting the need for better open-access resources.**
4. **Evaluation Limitations: Automated metrics like SARI and D-SARI may not fully capture human-perceived readability and coherence.**

Techniques Used

1. **Progressive Simplification:**
   - **Simplification occurs in a structured manner: discourse-level, topic-level, and lexical-level simplifications.**
   - **Iterative adjustments refine the outputs at each level.**
2. **Prompt Engineering:**
   - **Carefully designed prompts guide LLMs through the simplification process using ICL and COT techniques.**
3. **Evaluation Metrics:**
   - **SARI and D-SARI: Assess sentence and document-level simplifications.**
   - **BARTScore: Measures fluency and content preservation.**
   - **Flesch-Kincaid Grade Level (FKGL): Evaluates readability improvements.**
   - **GPT-Evaluation: Uses AI-based human-like evaluation to compare simplified outputs.**
4. **Over-Generate-Then-Filter:**
   - **Generates multiple outputs and filters them based on predefined rules to ensure adherence to requirements.**

Future Work:

1. **Multilingual Simplification: Expanding ProgDS to handle text in multiple languages.**

2. **General Text Processing Tasks: Applying ProgDS to other long-document tasks, such as summarization or translation.**

3. **Improved Framework Efficiency: Addressing computational complexity for handling longer documents efficiently.**

4. **Human Feedback Integration: Incorporating human feedback into the iterative process for further refinement.**

5. **Advanced Prompting Techniques: Enhancing prompts for improved output consistency across tasks.**

Beyond Sentence-level Text Simplification: Reproducibility Study of Context-Aware Document Simplification

Objective: This paper focuses on simplifying entire documents instead of individual sentences, aiming to keep the overall structure and flow intact. Traditional methods that simplify one sentence at a time often fail to maintain coherence across a document. To address this, the authors tested whether using a step-by-step "plan" to guide simplification improves results or not. They used an open-source dataset called **Wiki-auto**, instead of **Newsela-auto dataset**, to explore these ideas.

About the Datasets

Datasets Used: **Wiki-auto**: This dataset is made up of pairs of complex and simplified sentences and paragraphs from English Wikipedia and Simple English Wikipedia. Comparison: Wiki-auto has more examples of sentences being deleted or rephrased, while Newsela-auto focuses more on splitting sentences into simpler ones.

Models Used:

The paper tested three types of models:

1. Text-Only Models: These models (like **BART** and **LED**) work at different levels: sentence, paragraph, or entire document.
2. Context-Aware Models: These models (like **ConBART**) consider the surrounding sentences to improve simplifications.
3. Plan-Guided Models: These models use a predefined "plan" (like whether to **copy, delete, or split**) to guide how the simplification is done.

Experiments and How They Were Evaluated

1. How the Models Were Measured:
   - **SARI**: Checks if the output adds, keeps, or deletes the right parts of the text.
   - **FKGL**: Looks at how easy the simplified text is to read by performing calculations.

- **SMART** and **BARTScore**: Measure how well the simplified text keeps the original meaning and fluency.

2. What They Tested:
    - They compared models trained on the **Newsela-auto dataset** to see how well they worked on the **Wiki-auto dataset.**
    - They also tested whether including context or using a plan improved the results.

Result/Consclusion

1. Models that follow a simplification "plan" did the best, especially in making the text simpler and easier to read.
2. Models that use surrounding sentences (context-aware models) didn't always do better than simpler models, likely due to differences in datasets.

Issues:

1. Dataset Problems: Wiki-auto's quality is lower because it was created automatically, unlike Newsela-auto, which was manually created.
2. Model Shortcomings: Some models, like LED, often created overly long outputs that didn't match the simpler references.

Future Scope

1. Improve the quality of open-source datasets like Wiki-auto to make them more reliable.
2. Expand these methods to work for other languages or specific types of texts (e.g., **medical** or **legal** documents).