Transformers for Markdown Article Rewriting

Scott Hickmann
hickmann@stanford.edu

Problem

- Articles are long to read, hence it can be a huge productivity gain to summarize and rewrite an article
- State of the art paraphrases and summaries from transformer-based encoder-decoder models
- Improper markdown (MD) and HTML parsing from T5 and BART transformers
- Yet text markup is key to make an article easier to digest
- Markup is very present online, for example on Medium

Examples of problems:

- "A flock of ==frogs== were ==roaming== around the park in search of water once more." becomes "A flock of ==frogs===roaming===roaming===roaming====roaming====roaming====roaming..."
- "Once, a group of **frogs** were **roaming** around the forest in search of water." becomes "A flock of **frogs** were **roaming** around the park in search of water once more." whereas the same sentence without the markdown "**" syntax would become "A herd of frogs were wandering around the woods in search of water".

Methods

Paraphrasing

- Scrape MD sentences
- Paraphrase them without MD, then ask a human to add the MD back
- Encode all MD from both original and human generated text using (MD*n*) tags, with *n* the *n*-th markdown inline block
- Fine-tune the T5 model pretrained on sentence paraphrasing to preserve the syntax of these (MD*n*) tags

Summarization

- Scrape Medium articles, seperate into shorter parts
- Paraphrase each part without MD, then ask a human to add the MD back
- Encode all MD from both original and human generated text using (MD*n*) tags, with *n* the *n*-th markdown inline block
- Fine-tune the BART model pretrained on summarization to preserve MD syntax

Example

- Input with MD (fine-tune source): "Once, a group of **frogs** were **roaming** around the forest in search of water."
- Output without MD: "A herd of frogs were wandering around the woods in search of water"
- Human (fine-tune target): "A herd of **frogs** were **wandering** around the woods in search of water"

Evaluation

Using raw markdown articles that have been reserved for evaluation, we divide it into m subparts that are each get summarized or paraphrased individually. We remove all markdown and feed that through the original model without any fine-tuning (text 1), and feed the markdown encoded version through the final model with fine-tuning then remove the markdown (text 2). We then compute ROUGE 2 (*R*) and ROUGE L (*L*). These scores evaluate how well the original summaries and paraphrases are preserved after fine-tuning, but not how well the markdown syntax is.

Therefore, let's compute a score to evaluate the success of the markdown. As there is no standard metric for that, I developed and used the following custom metric. Let's define the functions *f*, which computes an enclosure check (makes sure that the output is surrounded by (MD0) tags which should be present based on how the MD encoder and decoder work), and *g*, which computes sentence similarity between two phrases by taking the average of the word embeddings of all words in the two phrases, and calculating the cosine similarity between the resulting embeddings:

$$f(\boldsymbol{v}) = \begin{cases} 1 & \text{if } \boldsymbol{v} = \text{"(MD0)\{entire content\}(MD0)"} \\ 0 & \text{otherwise} \end{cases} \qquad g(\boldsymbol{v}, \boldsymbol{w}) = \frac{\operatorname{avg}(\boldsymbol{v}) \cdot \operatorname{avg}(\boldsymbol{w})}{\|\operatorname{avg}(\boldsymbol{v})\| \|\operatorname{avg}(\boldsymbol{w})\|}$$

Let's define *M*, the markdown similarity score between the original text and the paraphrased/summarized text and *S* the overall score. Let's create ***v***, a 3D array containing the markdown tag contents for each (MD*n*) tag pair for each defined *n* for each paraphrased/summarized MD-encoded text. Let's also create ***w***, a 3D array containing the markdown tag contents for each (MD*n*) tag pair for each defined *n* for each original MD-encoded text (before paraphrasing/summarizing).

$$M = \frac{1}{m} \sum_{i=1}^{m} \left(\frac{1}{\operatorname{len}(\boldsymbol{v}_i)} \left(f(\boldsymbol{v}_{i,1,1}) + \sum_{j=2}^{\operatorname{len}(v_i)} \left(\frac{1}{\operatorname{len}(\boldsymbol{v}_{i,j})} \sum_{k=1}^{\operatorname{len}(v_{i,j})} \max\{g(\boldsymbol{v}_{i,j,k}, \boldsymbol{w}_{i,j,l}) : l = 1..\operatorname{len}(w_{i,j})\} \right) \right) \right)$$

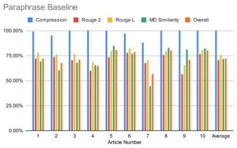
$$S = \frac{R + L + 2M}{4}$$

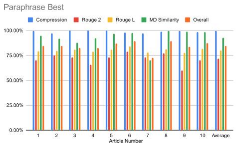
End-to-End

- Divide article into subparts, categorizing into paragraph vs. other (image, code, figures, etc.).
- Encode the markdown in paragraph subparts to (MD*n*) tags.
- Pass that into the BART fine-tuned summarizer model.
- Pass the result into the T5 fine-tuned paraphrasing model.
- Decode the resulting text from (MD*n*) tags to actual markdown.
- Combine all subparts back into a single document.

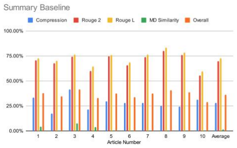
Results

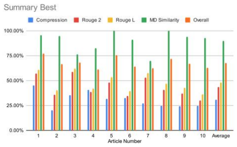
Paraphrasing





Summarization





Analysis

The model fails most often when there is a lot of nested markdown, which is particularly present at the footer of articles when referencing the author(s). We might want to add more fine-tuning data regarding footer text to address this issue. Overall, the results found are very promising considering the scarcity of data (manually collected) to achieve proper article rewriting. Collecting more data would definitely improve the performance especially when it comes to the ROUGE score of summarization, which is an inherently harder task than paraphrasing text.

Conclusion

The methods developed for this project have demonstrated their performance when it comes to simplifying or generating new and unique article rewrites.

References

- Ramsri Goutham. *High-quality sentence paraphraser using Transformers in NLP*. Sep 2021.
- Sam Shleifer. *Distilbart CNN 6-6*. Sep 2021.
- Chin-Yew Lin. *ROUGE: A Package for Automatic Evaluation of Summaries*. 2004.
- Yves Peirsman. *Comparing Sentence Similarity Methods*. May 2018.