Visualizing and Enforcing Order in Academic Literature Summarization

Sicong Liu **Zhengzhe Yang** **Zhaomin Zheng**
Language Technologies Institute
Carnegie Mellon University
{sicongli,zhengzhy,zhaominz}@cs.cmu.edu

Abstract

In most standard summarization tasks, the order of the summarization output does not matter across sentences. However, there are still cases where there needs to be a logical order between the summarization output sentences. In this paper, we attempt to address this particular sentence ordering issue that occurs in summarization. In particular, we replicate a study done on academic literature summarization [15] using the *arXiv* dataset [3], and focus on improving the generated summarizations by appending a new position decoder module to existing models. We propose a new way to incorporate the ROUGE evaluation metric for sentences with strict orders. We evaluate and benchmark various existing models and our transformer-based model on our custom metrics in order to determine their effectiveness. In conjunction, our visualizations across the different models help us explore model interpretability.

1 Introduction

Text summarization is the task of distilling a longer body of text into a shorter one, usually with the shorter sequence preserving the main content from the longer sequence. Traditionally [2][1], summarization have been categorized into single-document summarization and multi-document summarization. In this paper, we will solely focus on single-document summarization, as this is the format of our main dataset.

Furthermore, summarization can also either be *extractive* (identifying the most important sentences in a text and selecting them in a verbatim manner) or *abstractive* (producing entirely new sentences based on information in the source document) [2]. The position module we introduce is agnostic in terms of the method used, but since the study we replicate is done using an extractive methodology, we will also be focusing on comparing and evaluating based on extractive models.

We hypothesize that the current method of generating academic literature summaries is flawed because there lacks a logical order within the summary document. In fact, current techniques simply place the sentences in the order of descending confidence score, up to a certain threshold. Evaluating generated summaries is a challenging task in itself, with most research on summarization using ROUGE scores as the sole metric. We propose a new evaluation metric that places special emphasis on sentence ordering, and further aim to explain the current model outputs with visualizations.

2 Related Work

2.1 Early Work

The earliest work done on summarizing scientific documents was perhaps by Luhn in his 1958 work *The Automatic Creation of Literature Abstracts* [8]. He proposed entirely statistical approaches such as using word frequency and phrase frequency to extract the most salient sentences in an extractive way. In contrast, machine learning-based methods for extractive summarization occured much later on in the early 2000s [9][12], and instead used various traditional classifiers such as Naive Bayes to perform the same task.

2.2 Modern Approaches

Summarization systems were also recommended to take advantage of the inherent structure within journal articles, as they are all expected to contain introductions, related works, methodologies, results, and conclusions [10]. This was an approach taken by a work [4] that directly uses section information as categorical features, and achieves competitive results. Another work [3] proposes using a hierarchical encoder at both word and section levels to perform summarization. To generate a working label set, [5] uses a greedy heuristic-based approach to obtain an oracle set that best matches the article abstract. They [5] also provided a general set of hyper-parameters that many later papers such as [15] were based upon.

Outside of scientific documents, method such as LSTM-CNN based methods [11], pretraining transformer-based methods [16] and multi-level memory networks [6], have all been shown to work reasonably well for summarization tasks as well.

3 Data

3.1 The Original Dataset

The original dataset used in this project was compiled by Cohan et al. in [3] using academic papers they obtained from http://arxiv.org/. The published dataset itself consists of a pre-determined training, validation, and test partition. These partitions further consist of ArXiv articles in JSON form, each separated by newline characters. Features that are relevant towards the task are stored in these JSON data objects, and their schema is shown in Appendix B.

3.2 Data Preprocessing

It is important to first clarify our objective in order to determine the best way to preprocess this dataset: we wish to use text from the main article to select sentences that best match the text in the abstract.

Preprocessing text from the article itself is generally straightforward. We focus on tokenizing English sentences and pruning away unwanted parts.

Determining the best match for the abstract can be done in one of two ways:

1. Measuring the word overlap as a percentage and finding sentences that maximize this value.
2. Pre-constructing an alternative version of the abstract that only consists of sentences that originate from the main article text. In this case the best match would also be perfect matches. This best-match version is called *Oracle* in the original paper.

A greedy algorithm is presented in the paper [15] to generate abstract labels used for training using the second method. The labels are indicator vectors of a document, composed of 1s and 0s. Having a 1 in that position means the sentence in that position is present in the gold standard abstract. Such preprocessing could make the loss computation very easy with our predicted logits. An implementation of this algorithm could be found in our provided code repository[1].

The preprocessing steps produces files `inputs/*.json` (model inputs), `labels/*.json` (used for model training), and `human-abstracts/*.txt` (used for ROUGE evaluation). Their schemas are shown in Appendix C.

[1] https://github.com/scott0123/litsumm

3.3 Golden Set Re-generation

The original golden set, which is generated by a greedy algorithm described in Section 3.2, does not include any order information; instead, the golden set simply mark the sentences in a document to be 1 if it exists in the abstract, and 0 otherwise. In order to preserve the sentence order, we are motivated to re-generate the golden set using a beam search algorithm. This let's us find the abstract with the highest score while keeping the sentence in order. The newly generated label will reflect this fact by marking the sentences with sequence ordering instead of binary flags.

For each sentence in the human abstract, we find the sentence from the document with the highest ROUGE score in a beam search way. A detailed description of the algorithm could be found in Appendix D. With this newly generated golden set, we could train our model with an ordered sequence.

4 Methodology

4.1 Overview

The overall model architecture consists of an encoder and a decoder, just like many other literature summarization methodologies. However, we present a number of variants of these sub-components. In a nutshell, following Xiao and Carenini [15], we again present the document encoder that could capture the local contexts and topic information from a document in Section 4.2.2. In addition, we present a transformer-based encoder (Section 4.2.3 that we hope could leverage the multi-head attention mechanism to better capture context-based information. For decoders, in addition to the two kinds of attention-based and concatenation-based decoders (Section 4.3.2 and Section 4.3.1), we also present a sentence-wise decoder (Section 4.3.3) with the advantage of taking an ordered sequence version of the abstract into consideration.

We also propose a new sentence-wise ROUGE metric: predicting the wrong order for a document would now result in a penalty in evaluation. In this case, the new evaluation metrics in Section 4.4 would perform a one-to-one matching with the sentence with the ground truth human abstract.

4.2 Encoder

4.2.1 Sentence Encoder

After all the words are converted to embeddings from GloVe, we simply compute the average embedding for a sentence using words w from the sentence S with embedding function $\mathcal{F}_{emb}$, that is:

$$E_{\text{sentence}} = \frac{1}{|S|}\sum_{w\in S}\mathcal{F}_{emb}(w) \tag{1}$$

In our modeling, the word embeddings are frozen, that is, the words won't be updated during back-propagation.

4.2.2 Document Encoder

The objective of the document encoder is to transform the sentence embeddings learnt in Section 4.2.1 to three representations for each sentence, including sentence representation, document representation and topic segment representation. The transformations are done with a single bi-directional GRU model that takes all sentence embeddings of a document as input. The topic segment representation captures local context information while the document representation captures global context information [15].

Sentence representation is simply the concatenation of the forward and backward hidden states of that sentence, that is:

$$sr_i = (hidden_i^f : hidden_i^b) \tag{2}$$

Document representation is a concatenation of the final forward hidden state and the final backward hidden state, which is the same for all sentences in the same document. It can be described mathematically as:

$$d = (hidden_n^f : hidden_0^b) \tag{3}$$

Topic segment representation is calculated using the LSTM-minus method [14], which represents a segment using the difference between the end hidden state and the start hidden state of the segment. Similar to the above two representations, topic segment representation is the concatenation of the forward subtraction and the backward subtraction, which is the same for all sentences in the same topic segment. In the current task, topics are determined by the sections of a document, such as Introduction, Related work, etc. Topic segment representation can be mathematically described as:

$$l_t = (hidden^f_{end_t} - hidden^f_{start_t - 1} : hidden^b_{start_t} - hidden^b_{end_t + 1}) \tag{4}$$

Where sr represents the sentence representation, d represents the document representation, and l represents the topic segment representation. i represents the index of any sentence and n represents the index of the last sentence in a document.

4.2.3 Transformer-based Encoder

The transformer-based encoder [13] is used as an alternative to the Document Encoder. Since both kinds of encoders may have varying degrees of success in capturing local and global context, it is only natural to also include the transformer encoder in our model.

4.3 Decoder

4.3.1 Concatenation-based Decoder

After the document has been transformed to the aforementioned three levels of representations, the Concatenation-based Decoder simply concatenates them and feeds them to a multi-layer perceptron to make the final prediction. No further transformation or attention is computed based on this concatenated form.

4.3.2 Attentive-context Decoder

For the Attentive-context Decoder, instead of simply concatenating the sentence representation, the document representation, and the topic segment representation, we could compute a series of attention scores to use as context weights, which we mathematically describe as:

$$\text{score}^d_i = v^T \text{tanh}(W_a(d : sr_i)) \tag{5}$$

$$\text{score}^l_i = v^T \text{tanh}(W_a(l_t : sr_i)) \tag{6}$$

$$\text{weight}^d_i = \text{softmax}(\text{score}^d_i) \tag{7}$$

$$\text{weight}^l_i = \text{softmax}(\text{score}^l_i) \tag{8}$$

$$\text{context}_i = \text{weight}^d_i \cdot d + \text{weight}^l_i \cdot l_t \tag{9}$$

$$\text{input}_i = (sr : \text{context}_i) \tag{10}$$

$$h_i = \text{dropout}(\text{ReLU}(W_{mlp}\text{input}_i + b_{mlp})) \tag{11}$$

$$p_i = \sigma(W_h h_i + b_h) \tag{12}$$

Where sr represents the sentence representation, d represents the document representation, and l represents the topic segment representation. The output p represents the confidence score for each sentence in the document.

4.3.3 Position Decoder

Position decoder is a new component introduced as part of our research. The purpose of this decoder is to assign an position confidence value to each selected sentence so as to ordered them.

The architecture of position decoder is a simplified version of concat-based decoder. The difference between this and the concat-based decoder is that the input of position decoder is only the sentence encoding.

To train the position decoder, we used a transfer-learning mechanism. We first trained the encoders and a concat-based decoder as in the original paper. With the encoders weights frozen, we then use the selected sentences in the new golden set as input and normalize their labels as output. As a result, the trained decoder can predict an position confidence between 0 to 1 for every selected sentence. The selected sentences can then be ordered according to the confidence.

At inference stage, the trained sentence order decoder can be appended at the end of any of the other models.

4.4 Evaluation Metrics

4.4.1 ROUGE

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It is a widely-used metric to automatically evaluate the quality of a machine-generated summary compared to a human summary reference [7]. The main idea of ROUGE is to measure the recall of summarization by calculating the percentage of words or n-grams in the human summary appear in the machine-generated one. Although ROUGE can effectively measure wording similarity between the machine-generated summary and the human reference, it does not take into account the order of words and n-grams. Therefore, when evaluating a relatively long summary, like in this literature summarization task, ROUGE cannot measure whether the ordering of sentences is reasonable.

In the original paper, the machine-generated summary is constructed by selecting sentences with the highest confidence until an artificial word limit of 200 is met. The selected sentences are then joined with spaces. Afterwards, the concatenated summary is evaluated against the original human-written abstract using ROUGE. As we discussed above, ROUGE is decided by words and n-grams alone without any additional ordering information. Therefore, changing the ordering of the selected sentences does not affect the evaluation result of the original ROUGE. We also conducted several experiments to confirm this speculation.

However, in literature summarization, an abstract is usually a paragraph containing several sentences, the ordering of which affects the logical organization of the paragraph and thus affects the quality of the generated abstract. Therefore, we deem the inability to measure the ordering of sentences as a shortcoming of the original evaluation method. In our research, we propose a sentence-wise ROUGE method to overcome this problem.

4.4.2 Sentence-wise ROUGE

We propose a sentence-wise ROUGE method to evaluate the quality of the generated summary taking into account the order of the sentences.

More specifically, we apply ROUGE between every pair of sentences between the generated abstract and the human abstract and take an average. If the generated abstract has more sentences than the human abstract, we simply discard them. If the generated abstract has fewer sentences than the human abstract, we append empty strings to fill in the blanks.

5 Experiments

5.1 Model training

The data consisted of a training, validation, and test set; which had a size of $201,427$ documents, $6,436$ documents, and $6,431$ documents respectively. During training, two sets of labels are used, which are generated from the greedy algorithm from [15] and Section 3.3 respectively.

Given the expensive computation and the size of our training set, to minimize the training time, we resorted to a much higher initial learning rate under a scheduler that gradually decay. This worked well and save potentially hundreds of GPU-hours. Other hyperparameters are chosen based on [15], [5], as well as our own experience.

After a forward pass, each sentence in a document would be assigned a logit, which corresponds to the probability of being a part of the predicted extractive abstract. We use the logit instead of the direct probability for computational stability. Binary cross entropy loss was used as the loss function. During inference, two decisions need to be made in order to achieve a high score in evaluation: for each sentence, our model needs to decide whether to keep this sentence to be our predicted abstract, and at which position should this sentence be. Given the fact that we are evaluating the predictions sentence-wise, the predictions that have the incorrect ordering would not achieve a high score. This is a step forward comparing to other literature summarization since traditional ROUGE evaluation does not take ordering of sentences into consideration.

For our results, the evaluation of the two golden set will be reported, which we call the Oracle score. The Oracle score serves as the theoretical limit of our model for the evaluation sets, since the model is trained based on the Oracle labels. Then, the results for the actual predictions are evaluated again using the Sentence-wise evaluation metric. The Full report of results could be found in Section 6.

5.2 Golden Set Validation

Since we generated a new set of golden labels based on the sentence-wise matching with the human abstracts and devised a new evaluation mechanism that takes advantage of the ordering information, we would evaluate the old labels and the newly generated labels with this new evaluation metric (Section 4.4). This is done to show that using the newly form golden set is indeed an effective way to verify ordering during the evaluation phase. These two sets of labels, which are generated from the greedy algorithm from [15] and Section 3.3 respectively, are evaluated against the human abstracts from the actual documents using the Sentence-wise ROUGE from Section 4.4.2. We are presenting these results in order to 1) give an upper bound of the theoretical performance for our task and 2) demonstrate that our training with the re-generated golden set is effective at enforcing the order information for sentences.

5.3 Transfer Learning

We train the model with these two different sets of golden labels. The decoder of our model will assign a confidence score to each sentence in the document, and these scores could help us identify the ones that should exist in the final abstract prediction. If the model is predicting without ordering, the output order of the sentences simply depends on the confidence score of each sentence, ranking the sentences with its corresponding confidence score from high to low, until a word limit is reached. If the model is indeed predicting the ordering of the sentences in the output abstract, we sort the sentence based on this ordering information. Later, we evaluate it using the sentence-wise evaluation metric proposed by us from Section 4.4. The training of order prediction could be categorized as a transfer learning model, in which we use the pre-trained model at hand and continue to train it with addition information, in our case, the ordering information for each document.

6 Results

6.1 Model Evaluation

6.1.1 Original ROUGE Evaluation

From Table 1, we can have a general idea about how the model is doing with its old golden set and the old evaluation metrics. To our surprise, the one with concatenation decoder, which is much simpler than the model with attentive decoder, achieved the best performance. However, we cannot conclude that the concatenation decoder was better than the attentive one because we only trained them for less than 5 epochs without waiting for saturation, due to time and computational resources limitation. So the attentive model may possibly surpass the concatenation one if they were both fully trained. Instead, we can safely conclude the concatenation model had a faster saturation rate, because of its simpler model structure and fewer number of parameters.

6.1.2 Sentence-wise Golden Set Validation

Table 2 reports the evaluation of the two Oracle Golden sets against the human abstracts using Sentence-wise ROUGE. From our results, apparently the Golden set generated by our algorithm (Section 3.3) outperforms the Golden set provided by [15] a lot. This is because they did not try to incorporate the ordering of the sentences, and the evaluation did not either. Therefore, their evaluation is much more lenient than ours, leading to a sub-optimal abstract generation. In our case, we push the abstract predict to penalize on the abstract with the incorrect ordering by evaluating the sentences one by one. The detailed description of this sentence-wise evaluation could be found in Section 4.4.

	Original ROUGE		
Model	ROUGE-1	ROUGE-2	ROUGE-L
Baseline	45.37	16.98	29.90
Attentive context	45.57	17.22	29.95
Concat	**46.26**	**17.95**	**30.63**
Transformer	41.25	13.2	27.19
Old Oracle	51.71	21.23	32.33

Table 1: The performance of the models using original ROUGE evaluation The authors' are in the first block while ours are in the second block. The oracles correspond to using the ground truth labels, introduced in Section 3.3.

	Sentence-wise ROUGE		
Model	ROUGE-1	ROUGE-2	ROUGE-L
Old Oracle	17.20	5.07	16.98
Sent-wise Oracle	39.87	24.19	39.36

Table 2: The results using the sentence-wise ROUGE evaluation metric. This evaluation will enforce the ordering of the sentences which would push our model to predict the sentences to represent the abstract as well as the correct ordering.

6.1.3 Sentence-position Transfer Learning Evaluation

For each set of golden labels, we output 2 sets of results of whether the ordering is also predicted, constituting a total of 4 sets of results. The detailed results reporting could be found in Table 3. As we can see, the model we trained with the golden set we created outperforms the model trained with old oracle labels. The score is much better with the transfer learning technique, since the ordering information should be learned during training. These results have proven that the labels we generated could indeed reflect the ordering information and training with ordering information is necessary to achieve a relatively higher score with our sentence-wise evaluation.

	Sentence-wise ROUGE		
Model	ROUGE-1	ROUGE-2	ROUGE-L
Old	10.27	1.50	11.38
Sent-wise	20.66	5.30	20.73
Old-Order	14.92	2.51	15.70
Sent-wise-Order	**21.45**	**5.65**	**21.44**

Table 3: Old model was trained with the original Oracle labels while Sent-wise model was trained with new Oracle labels. These two models are evaluated by the order of the confidence value of the predicted sentences. Old-Order model was learned by transfering the Old model onto order prediction problem. Similarly, Sent-wise-Order model was learned by transfering the Sent-wise model onto order prediction.

6.2 Error Analysis

We visualize the sentence-level confidence scores for a given article. In this case we randomly selected the first article in our test set which happened to be the article with arXiv ID `cs0003081`. This is illustrated in Figure 1.

We see that this paper has four sections, this is represented in the subplots of this figure as the vertical axis. If we were to find this paper back in the original arXiv context, we see that section 0 would correspond to the INTRODUCTION section, section 1 the MODELLING VARIABLE WORD RATES section, section 2 the VARIABLE WORD RATE LANGUAGE MODELS section, and section 4 the CONCLUSION section. The subplot on the bottom is the *oracle* labels, which represents what each model is trying to learn.

As we can see, each of the model places a different emphasis (confidence) for each sentence in each section. The *GRU-base* model found the beginning of the conclusion section as good candidates for its abstract, while the *GRU-concat* model found the middle of the introduction section as good candidates. This can be contrasted with the *GRU-attentive* model, which found the final part of the introduction section indicative, as well as the beginning of the conclusion section. Finally, the *transformer* model displayed similar characteristics as the *GRU-attentive* model.

While it may be quite harsh, if we compare the various model outputs with the *oracle* labels, we evidently see that the models in their current generation are still not capable of picking out the subtleties of the characteristics of a sentence that makes the best candidate for forming an abstract. This shows that there is still much work to be done in this field.

We supplement this figure with a comparison of the actual human-readable outputs for each model type that we had investigated. These are all listed in appendix A.

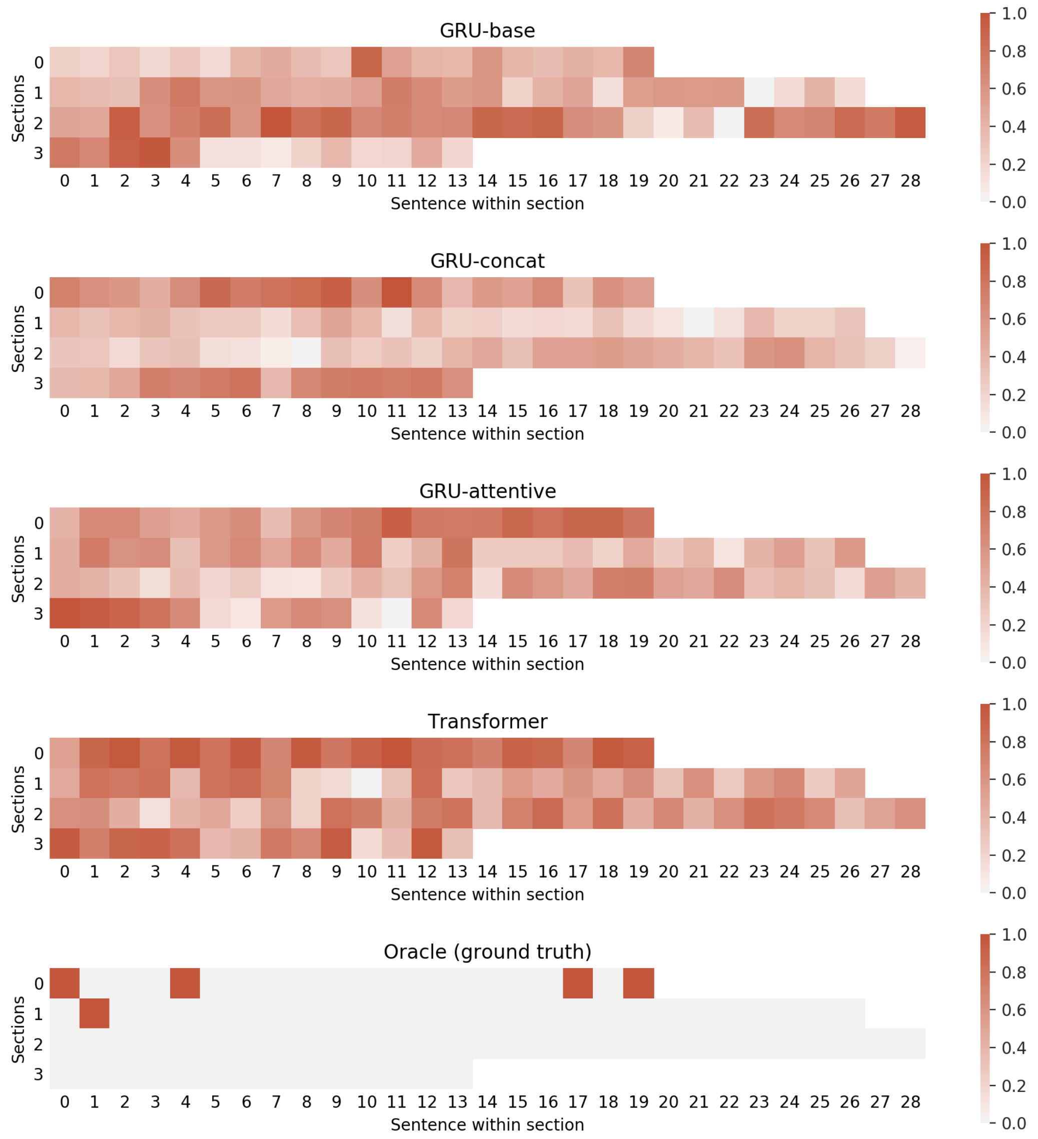

Figure 1: Sentence-level confidence scores across various models

We also manually analyzed 10 examples with the highest ROUGE-2 score and the 10 examples with the lowest. From our observation, most of the sentences that are parts of the prediction are selected from the *Introduction* and *Conclusion* / *Summary* sections, as one might expect. This agrees with human intuition about abstract writing: an abstract should tell a summarized idea about the whole article, which is exactly the purpose of introductions and conclusions.

As for the errors, we noticed that for the manually analyzed samples, while one sample has an article in which the body and the abstract are in different languages, most of the samples have a prediction score that is aligned with their oracle score. In other words, the quality of the prediction is bounded by the quality of the oracle. If the human abstract is similar to some sentences in the article, we could end up with pretty good predictions that has a high ROUGE-2 score with the human abstracts. If not, which means the human abstracts are highly abstractive and cannot simply be represented using the sentences from the article itself, the performance of our model will thus be hindered. This motivates us to randomly sample 100 articles and plot their prediction evaluation and oracle evaluation. As from Figure 2, by randomly sampling 100 articles and sorting the prediction F1 scores, the oracle score is proved to be mostly correlated with the predictions, substantiating our speculation that the predictions are bounded by the best *oracle*.

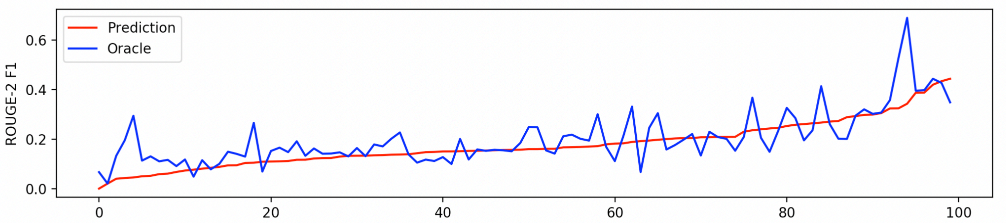

Figure 2: Randomly sampled 100 articles (oracle vs. predictions) (sorted)

6.3 Future Work

Although the sentence-wise ROUGE proposed in this project improved vanilla ROUGE on sentence ordering evaluation. It did not handle the situation where the predicted summary and the human summary may have different number of sentences in a truly sound way. This situation is actually very common in extractive summarization tasks. As the sentences from the original article can have various lengths. In current version of sentence-wise ROUGE, we simply remove excessive sentences when the generated summary has more sentences, which results in information loss, and add empty strings into the evaluation function when the generated summary has fewer sentences. These may not be fair comparisons. In the future, we propose to try a weighted-sum sentence-wise ROUGE, which calculates ROUGE scores between every two pair of generated summary sentence and reference summary sentence, with the final score bring the total sum. By assigning higher weights to sentence pairs that have similar relative position in their summary, and lower weights to other pairs, we may be able to fairly evaluate the ordering of the sentences without adding or taking away information.

7 Conclusion

In the paper, we replicated previous studies on academic literature summarization and evaluated an assortment on models on the traditional ROUGE metric. We proposed an alternative way to improve the generated summarization of any model by appending a new position decoder module to existing models. We then proposed a new sentence-wise ROUGE metric that better captures the inherent logical order within abstracts, and demonstrated that our proposed method of learning and enforcing a partial order in the summarization yielded a significantly better score according to this new metric. In addition, we illustrate the sentence-level confidence score of the existing models using a heat-map, and argue that the current generation of models are still far from optimal. Our results suggest that future work done in this field should also be wary of sentence order in the generated text, and possibly use our proposed metric as a way to help evaluate whether a logical order has been successfully preserved.

Appendix A Human-readable Model Outputs

Human abstract (article cs0003081)
the rate of occurrence of words is not uniform but varies from document to document . despite this observation , parameters for conventional @xmath0-gram language models are usually derived using the assumption of a constant word rate . in this paper we investigate the use of variable word rate assumption , modelled by a poisson distribution or a continuous mixture of poissons . we present an approach to estimating the relative frequencies of words or @xmath0-grams taking prior information of their occurrences into account . discounting and smoothing schemes are also considered . using the broadcast news task , the approach demonstrates a reduction of perplexity up to 10% . (0,0) (0,0)

Oracle (ground truth for training) (article cs0003081)
in both spoken and written language , word occurrences are not random but vary greatly from document to document . much of this work has evolved around the use of (a mixture of) the poisson distribution @xcite . . the constant word rate assumption is then eliminated , and we introduce a variable word rate @xmath0-gram language model . an approach to estimating relative frequencies using prior information of word occurrences is presented . the approach demonstrates the reduction of perplexity up to 10% . we show that the word rate is variable and may be modelled using a poisson distribution or a continuous mixture of poissons .

GRU-base (article cs0003081)
information from different language model statistics (, a general model and/or models specific to each topic) are then combined using methods such as mixture modelling @xcite or maximum entropy @xcite . let @xmath20 denote a relative frequency after we observe @xmath21 occurrences of word @xmath22 . the right figure demonstrates relative frequencies after a certain number of word occurrences . let @xmath33 denote a bigram entry (a word @xmath34 followed by @xmath22) in the model . a bigram probability @xmath37 may be smoothed with a unigram probability @xmath38 . using the interpolation method @xcite : @xmath39 where @xmath40 implies a “ discounted ” relative frequency (described later) and @xmath41 is a non - zero probability estimate (, the probability that a bigram entry @xmath33 exists in the model) . the difference was predictable because bigrams were orders of magnitude more sparse than unigrams . the approach demonstrated a reduction of perplexity up to 10% , indicating potential although the technique is still premature . because of the data sparsity problem , it is not clear if the approach can be applied to language model components of current state - of - the - art speech recognition systems that typically use 3/4-grams .

GRU-concat (article cs0003081)
recently , church and gale have demonstrated that a continuous mixture of poisson distributions can produce accurate estimates of variable word rate @xcite . lowe has introduced a beta - binomial mixture model which was applied to topic tracking and detection @xcite . although a constant word rate is an unlikely premise , it is nevertheless adopted in many areas including @xmath0-gram language modelling . in order to address the problem of variable word rate , several adaptive language modelling approaches have been proposed with a moderate degree of success . typically , some notion of “ topic ” is inferred from the text according to the “ bag - of - words ” model . the _ dynamic cache model _ k. sprck jones , s. walker , and s. e. robertson . a probabilistic model of information retrieval : development and status . technical report tr446 , university of cambridge , computer laboratory , 1998 . the beta - binomial mixture model for word frequencies in documents with applications to information retrieval . in _ proceedings of eurospeech-99 _ , volume 6 , pages 24432446 , budapest , september 1999 . interpolated estimation of markov source parameters from sparse data . in _ proceedings of the workshop : pattern recognition in practice

GRU-attentive (article cs0003081)
the _ dynamic cache model _ all the parameters of a language model may be completely determined according to probabilistic model of word rate , such as a poisson mixture . in this paper . the constant word rate assumption is then eliminated , and we introduce a variable word rate @xmath0-gram language model . an approach to estimating relative frequencies using prior information of word occurrences is presented . it is integrated with standard @xmath0-gram modelling that naturally involves discounting and smoothing schemes for practical use . using the darpa / nist 4e north american broadcast news task , in this paper , we have presented a variable word/@xmath0-gram rate language model , based upon an approach to estimating relative frequencies using prior information of word occurrences . poisson and negative binomial models were used to approximate word occurrences in documents of fixed length . using the broadcast news task , the approach demonstrated a reduction

of perplexity up to 10% , indicating potential although the technique is still premature . because of the data sparsity problem , it is not clear if the approach can be applied to language model components of current state - of - the - art speech recognition systems that typically use 3/4-grams .

Transformer (article `cs0003081`)
ir systems are typically based on unigram statistics (often referred to as a “ bag - of - words ” model) , coupled with sophisticated term weighting schemes and similarity measures @xcite . in an attempt to mathematically realise the intuition that an occurrence of a certain word may increase the chance that the same word is observed later , much of this work has evolved around the use of (a mixture of) the poisson distribution @xcite . lowe has introduced a beta - binomial mixture model which was applied to topic tracking and detection @xcite . several adaptive language modelling approaches have been proposed with a moderate degree of success . the _ dynamic cache model _ it is integrated with standard @xmath0-gram modelling that naturally involves discounting and smoothing schemes for practical use . using the darpa / nist 4e north american broadcast news task , in this paper , we have presented a variable word/@xmath0-gram rate language model , based upon an approach to estimating relative frequencies using prior information of word occurrences . interpolated estimation of markov source parameters from sparse data . in _ proceedings of the workshop : pattern recognition in practice

Appendix B Pre-processed Data File Format

```
{
    "article_id": "...",
    "article_text": [...],
    "abstract_text": "...",
    "labels": [...],
    "section_names": [...],
    "sections": [...]
}
```

Appendix C Data File Format before Pre-processing

```
# inputs/*.json
{
    "id": "...",
    "inputs":
    [{
        "text": "...",
        "tokens": [...],
        "sentence_id": "...",
        "word_count": "..."
    }, ...],
    "section_names": [...],
    "section_lengths": [...]
}


# labels/*.json
{
    "id": "...",
    "labels": [...]
}


# human-abstracts/*.txt
"..."
```

Appendix D Golden Set Re-generation Algorithm

Algorithm 1: Beam Search to re-generate the golden set sentence-wise

```
Result: The Golden Set based on the sentence ordering for the human abstract
beam_width = N;
beam_sequence = [([], 0.0)];
for all sentences : human abstract do
    temp_sequences = [];
    s1 = one sentence from a human abstract;
    ref_ind = the index to this reference sentence;
    for all sentences : document do
        s2 = one sentence from a document;
        s2_ind = index to this sentence in the document;
        ROUGE_score = get_ROUGE_score(s1, s2);
        for sequence, total_score : beam_sequence do
            new_sequence = copy(sequence);
            if s2_ind not in sequence then
                new_score = total_score + ROUGE_score;
                append s2_ind to new_sequence;
                append tuple(new_sequence, new_score) to temp_sequences
            end
        end
    end
    beam_sequences = the highest N elements from temp_sequences;
end
return the first tuple from the beam_sequences
```

References

[1] Mehdi Allahyari, Seyed Amin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys J. Kochut. Text summarization techniques: A brief survey. *ArXiv*, abs/1707.02268, 2017.

[2] Dipanjan Das Andr. A survey on automatic text summarization. 2007.

[3] Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. A discourse-aware attention model for abstractive summarization of long documents. In *NAACL-HLT*, 2018.

[4] Ed Collins, Isabelle Augenstein, and Sebastian Riedel. A supervised approach to extractive summarisation of scientific papers. *ArXiv*, abs/1706.03946, 2017.

[5] Chris Kedzie, Kathleen McKeown, and Hal Daumé. Content selection in deep learning models of summarization. In *EMNLP*, 2018.

[6] Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. Abstractive summarization of reddit posts with multi-level memory networks. In *NAACL-HLT*, 2018.

[7] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[8] Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2:159–165, 1958.

[9] Inderjeet Mani and Mark T. Maybury. Automatic summarization. *Computational Linguistics*, 28:221–223, 2001.

[10] Ani Nenkova and Kathleen McKeown. Automatic summarization. 2011.

[11] Shengli Song, Haitao Huang, and Tongxiao Ruan. Abstractive text summarization using lstm-cnn based deep learning. *Multimedia Tools and Applications*, 78:857–875, 2018.

[12] Simone Teufel and Marc Moens. Summarizing scientific articles: Experiments with relevance and rhetorical status. *Computational Linguistics*, 28:409–445, 2002.

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[14] Wenhui Wang and Baobao Chang. Graph-based dependency parsing with bidirectional lstm. pages 2306–2315, 01 2016.

[15] Wen Xiao and Giuseppe Carenini. Extractive summarization of long documents by combining global and local context. *arXiv preprint arXiv:1909.08089*, 2019.

[16] Haoyu Zhang, Yeyun Gong, Yu Yan, Nan Duan, Jianjun Xu, Ji Wang, Ming Gong, and Ming Zhou. Pretraining-based natural language generation for text summarization. *ArXiv*, abs/1902.09243, 2019.