Interpretable Multi-Headed Attention for Abstractive Summarization at Controllable Lengths

Ritesh Sarkhel* **Moniba Keymanesh*** **Arnab Nandi** **Srinivasan Parthasarathy**
Department of Computer Science and Engineering
The Ohio State University
sarkhel.5,keymanesh.1,nandi.9,parthasarathy.2@osu.edu

Abstract

Abstractive summarization at controllable lengths is a challenging task in natural language processing. It is even more challenging for domains where limited training data is available or scenarios in which the length of the summary is not known beforehand. At the same time, when it comes to trusting machine-generated summaries, explaining how a summary was constructed in human-understandable terms may be critical. We propose Multi-level Summarizer (MLS), a supervised method to construct abstractive summaries of a text document at controllable lengths. The key enabler of our method is an interpretable multi-headed attention mechanism that computes attention distribution over an input document using an array of timestep independent semantic kernels. Each kernel optimizes a human-interpretable syntactic or semantic property. Exhaustive experiments on two low-resource datasets in English language show that MLS outperforms strong baselines by up to 14.70% in the METEOR score. Human evaluation of the summaries also suggests that they capture the key concepts of the document at various length-budgets.

1 Introduction

Great progress has been made in recent years on abstractive summarization of text documents. Among existing works, sequence-to-sequence networks with attention (Gehring et al., 2017; Liu et al., 2018a) have been one of the clear front-runners. Being able to constrain the length of a summary while preserving its desirable properties has many real-world applications. One such application is content optimization for variable screen-sizes. Online content creators such as news portals, blogs, and advertisement agencies with audiences on multiple platforms customize their content based on display-area for best experience. However, there has not been much work on summarization at controllable lengths until recently. High variance in screen-sizes often require extensive human supervision to perform these modifications. As most sequence-to-sequence networks (Rush et al., 2015; Nallapati et al., 2016) do not enforce the length of a summary, for scenarios as mentioned above, one may need to employ an ensemble of networks to cover all possible lengths. There are two major challenges in following this approach for real-world applications. First, training sequence-to-sequence networks is a resource-intensive task (Strubell et al., 2019). To train a network for generating summaries budgeted at length b, we need a parallel corpus of text documents and their gold-standard summaries at length b. Constructing a large enough corpus with summaries budgeted at $b, \forall b \in (0, 1)$ may not be possible and/or cost-efficient for a number of domains. This is one of the main reasons why most existing works on abstractive summarization evaluate their model on large-scale news corpus datasets (Nallapati et al., 2016; Hermann et al., 2015), leaving out a number of important but low-resource domains (Magooda and Litman, 2020; Parida and Motlicek, 2019) where the number of available training documents is limited. Second, the range of possible length-budgets $\mathcal{R}(b)$ may not always be known beforehand. In many scenarios, it can be known as late as during run-time. Therefore, we formalize the summarization task addressed in this paper as follows.

Problem Definition: Given a document S of length N (tokens) and a maximum token budget of b, we aim to construct an abstractive summary s_b that satisfies the following conditions, **C1:** information

* The first two authors contributed equally to this work

Proceedings of the 28th International Conference on Computational Linguistics, pages 6871–6882
Barcelona, Spain (Online), December 8-13, 2020

Input text

police are hunting a man aged between 50 and 60 suspected of robbing a bank in broad daylight and running off with £3,000 in cash. the robbery took place at 12.30pm at a lloyds bank branch in fairwater, cardiff, police said. **detectives have issued cctv images of the suspect, who is 50 to 60, 5ft 9in to 6ft and was wearing black clothing. the white male suspect, who has greying black hair and wore glasses, was captured on camera inside the bank.** detectives said no one was injured during the robbery and they were 'confident' the public would be able to identify the suspect. detective sergeant andy miles, from fairwater cid, said: 'inquiries are continuing to identify the culprit. the cctv is clear and i am confident that members of the public will know his identity...'. (truncated)

Summary at compression budget = $\frac{1}{2}$

police are hunting a man aged between 50 and 60 suspected of robbing a bank in broad daylight and running off with £3,000 in cash. the robbery took place at 12.30pm at a lloyds bank branch in fairwater, cardiff, police said. the white male suspect, who has greying black hair and wore glasses, was captured on camera inside the bank. detectives have issued cctv images of the suspect, who is 50 to 60, 5ft 9in to 6ft and was wearing black clothing. detective sergeant andy miles, from fairwater cid, said: 'inquiries are continuing to identify the culprit.

Prototype Summary

robbery took place at 12.30pm at a lloyds bank branch in fairwater , cardiff. detectives have issued cctv images of the suspect , who is 50 to 60. detective sergeant andy miles , from fairwater cid , said : ' inquiries are continuing to identify the culprit.

Figure 1: MLS *expands the highlighted sentences in the prototype summary to the boldfaced tokens in the input text to construct a summary budgeted at half-length of the input text*

redundancy is minimized in s_b; **C2 :** coverage of the major topics of S is maximized in s_b; **C3 :** length of s_b is maximal within the specified budget b without adversely affecting the conditions C1 and C2 i.e., $|s_b| \leq b$ & $\nexists\, s_c$ such that $|s_b| < |s_c| \leq b$. C1 and C2 ensure that the properties of a high-quality summary is preserved in s_b, whereas C3 ensures that s_b is the largest possible summary that can be constructed within budget b without compromising its quality. Note that C1 and C2 are seemingly contradictory to each other as the length of the summary increases. Our goal is to find the optimal tradeoff.

Early works on incremental summarization (Buyukkokten et al., 2001; Yang and Wang, 2003) leveraged structural tags supported by document markup languages to generate summaries at various lengths, thus imposing a serious constraint on the document formats (e.g. XML, HTML) that come under the purview of such methods. Incremental sampling of sentences based on a salience score (Otterbacher et al., 2006; Campana and Tombros, 2009) can partially solve this problem by constructing extractive summaries of the input document. We show in Section 3 that these sampling-based methods often fail to preserve the desirable properties of a high-quality summary. Among recent works, (Kikuchi et al., 2016) were the first to propose a supervised method for controlling length during abstractive summarization. Their work was later extended by (Fan et al., 2017) who introduced the length of a summary as an input to the network. However, instead of exact input, they approximate the length to a predefined value-range, often failing to adhere to the allocated budget in a number of cases. (Liu et al., 2018b) address this issue by proposing a convolutional encoder-decoder network, introducing the desired summary length as an input to the initial state of the decoder. We compare and report its performance on two datasets in our experimental setup in Section 3.

Unfortunately, when it comes to interpreting[1] these models i.e. how the summaries came to be, the answer still remains illusive. Explaining how a machine-generated summary was constructed, has become a necessity under the newly introduced General Data Privacy Regulation Act (ITGP, 2017), especially for applications in enterprise (Sarkhel and Nandi, 2019; Keymanesh et al., 2020) and biomedical domain (Moradi and Ghadiri, 2018; Sarkhel et al., 2018). Some recent efforts have proposed using interpretable heatmaps (Baan et al., 2019) generated from the attention distribution over an input sequence for interpreting model behaviour. However, they are still quite limited (Jain and Wallace, 2019) in consistently explaining all aspects of a neural summarizer. This leaves a gap in the ongoing efforts (Song et al., 2020a; Song et al., 2020b) to generate abstractive summaries that are guided by human-interpretable semantic/syntactic qualities. Briefly, the main goal of attention mechanism in a encoder-decoder network is to assign a softmax score to every encoder hidden state (based on its relevance to the token

[1] "the ability to explain or to present in understandable terms to a human", (Doshi-Velez and Kim, 2017)

being decoded) and amplify those that are assigned high scores through a weighted average. Source-target attention (Nallapati et al., 2016) relies on another sequence for computing these scores, whereas self-attention (Vaswani et al., 2017; Paulus et al., 2018) operates over the elements in the current input sequence. A multi-headed attention mechanism allows a neural model to speed up training by enabling parallelization across timesteps. The number of operations in the computation of self-attention, however, scales quadratically with input length, making it a computationally expensive operation for long input sequences. Training such a network for a summarization task would require a large parallel corpus of input documents and their corresponding gold-standard summaries budgeted at b. The role of some of the attention-heads during abstractive summarization is also not transparent (Baan et al., 2019). To address these, we replace self-attention with a lightweight, interpretable alternative. Instead of projecting each input sequence multiple times[2] at every timestep, we encode an input sequence only once, using a timestep-independent kernel ($\vec{Q}$) learned in an unsupervised or distantly supervised way from the input document. Each kernel has a human-interpretable syntactic/semantic role. Every attention-head in this multi-headed mechanism computes an attention distribution over the input sequence using a unique kernel $\vec{Q}_i$, recycling it at every timestep. Compared to self-attention, our proposed attention mechanism scales linearly with the input sequence length and leverages significantly less number of trainable parameters. As we will show in Section 3, this allows us to train our network on limited training samples in low-resource datasets.

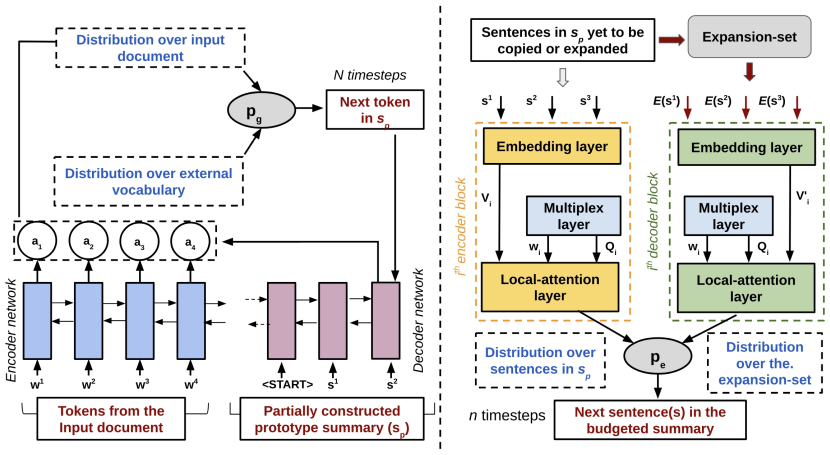

Figure 2: *An overview of* MLS *architecture. The PG-Network (left) constructs a prototype summary s_p from the input document. The Pointer-Magnifier network (right) constructs the length-constrained summary from s_p using interpretable sentence-level attention*

We propose MLS – a supervised method to generate abstractive summaries at arbitrary lengths in this paper. It computes a length-constrained summary s_b budgeted at length b by soft-switching between a copy and expand operation over a prototype summary s_p constructed from the document. The key enabler in this process is an interpretable, multi-headed attention mechanism. We develop a length-aware encoder-decoder network, called the *Pointer-Magnifier* network that leverages this attention mechanism to construct summaries within a specified length. We train our network on limited training samples from two cross-domain datasets: the MSR-Narrative (Ouyang et al., 2017) and Thinking Machines dataset (Brockman, 2018). Exhaustive evaluation on a range of success metrics shows that MLS performs competitively or better against strong baseline methods. Subsequent human evaluation of summaries generated by MLS suggests that they accurately capture the main concepts of the input document. To summarize, some of the major contributions of this work are as follows:

- We propose MLS, a supervised approach to generate abstractive summaries of a text document at controllable lengths.
- We develop a length-aware encoder-decoder network that leverages an interpretable, multi-headed attention mechanism to construct length-constrained summaries.
- Experimental results on two cross-domain datasets show that trained on limited training samples, MLS was able to generate summaries that are coherent and captured the key concepts of a document.

[2]one time each to compute the query, key and value matrix (Vaswani et al., 2017) from the input sequence

2 Proposed Methodology

MLS constructs a length-constrained summary of a document in two steps. First, it derives a prototype summary s_p from the document, covering its major concepts. Then, it expands or shortens it, depending on the length-budget to create the final summary. We employ a pair of encoder-decoder networks at both steps. For the first step, we extend the PG-network (See et al., 2017). We develop a length-aware encoder-decoder network for the second step. We describe both steps in greater detail in the following sections.

2.1 Generating the Prototype Summary

We extend PG-Network by (See et al., 2017) to construct the prototype summary s_p of a document. We tokenize the document and feed it to the encoder network sequentially. As the encoder hidden states are updated, the decoder network constructs the prototype summary one token at a time by soft-selecting between tokens in the input document and an external vocabulary. The decoding process is guided by an attention distribution[3] computed over the input document and the external vocabulary. An overview of this network is shown in Fig 2. We point the readers to the work by See et al. for more background on this network. An example prototype summary is shown in Figure 1. Contrary to existing prototype-text guided summarization methods (Liu et al., 2019; Saito et al., 2020), we do not specify the length of the prototype summary as an input of the network, rather infer it by outputting tokens until the EOS token is produced. We discuss the training and parameter settings of the network used in our experiments in Section 2.3. It is worth mentioning here that one of the main reasons to select the PG-Network as our architecture of choice for this step is due to its capability to construct a summary by looking up a learned language model. Other networks with similar capabilities can also be used, as this step has a transitive effect on the next phase of our approach.

2.2 Constructing the Length-Constrained Summary

To construct a summary within length-budget b, we develop the *Pointer-Magnifier* network: a length-aware, interpretable, encoder-decoder network. An overview of the network is shown in Fig. 2. It consists of a multiplex layer, an encoder (yellow rectangles) layer and a decoder (green rectangles) layer. The encoder layer takes the prototype summary constructed in the previous step as input. The decoder layer outputs the final summary. We describe each layer in detail below.

A. The Multiplex Layer and Interpretable Kernels: In an effort to build a transparent network, we embody three qualitative properties that are associated with a high-quality summary in our network. A high-quality summary, (1) maximizes the coverage of the *major topics* (Φ_1) and (2) *keywords* (Φ_2) appearing in the input document, while (3) minimizing the amount of *redundant information* (Φ_3). We encode each property using a semantic kernel ($\vec{Q}_i$), learned using an unsupervised or distantly supervised way from the input document itself. Every kernel plays a unique, human-interpretable syntactic/semantic role in constructing the final summary. One of the key components in this process is the multiplex layer $\mathbb{M}$. Physically, it is a nested matrix of dimensions 3×3 shared between the encoder and decoder layer. Each row in $\mathbb{M}$ contains the following information: (a) a distance-metric ($dist_i$), (b) a scalar value (w_i), and (c) a semantic kernel ($\vec{Q}_i$), where $-1 \leq w_i \leq 1, \forall i$ & $\Sigma_i^3 w_i = 1$. During inference, each of these kernels measures the contribution of every sentence in the prototype summary towards optimizing one of the properties Φ_i, $1 \leq i \leq 3$, mentioned above. w_i represents the relative weight assigned to the property Φ_i in constructing the final summary. We compute the kernels as a preprocessing step.

Defining the Kernels: To encode the property ϕ_1, we define $\vec{Q}_1$ as a matrix of dimensions 3×300, where each row of $\vec{Q}_1$ represents one of the three most dominant topic vectors of the input document as a 300-dimensional vector. We use an unsupervised LDA-based model (Blei et al., 2003) to derive these topic vectors. Symmetric KL-divergence is used as the distance metric ($dist_1$). Similarly, we encode the property ϕ_2 as a single dimensional vector $\vec{Q}_2$ of length 50, where each vector component represents the relative frequency of one of the 50 most frequent keywords in the input document. We use RAKE (Rose et al., 2010), a publicly available library to identify the keywords of a document.

[3] we closely followed the official implementation at: https://github.com/abisee/pointer-generator

Symmetric KL-divergence is used as the distance metric ($dist_2$). Finally, we encode ϕ_3 as a matrix $\vec{Q_3}$ of dimensions $p \times 300$, where the i^{th} row of $\vec{Q_3}$ represents an embedding of the i^{th} sentence in the input document. We compute the embedding vector of each sentence using a pretrained model (Le and Mikolov, 2014) on English Wikipedia corpus. Cosine similarity is used as the distance metric ($dist_3$). Our choice of unsupervised/distantly supervised kernels reflects our motivation (see Section 1) to leverage a limited number of training samples from the experimental dataset to construct the final summary. We discuss the role played by each semantic kernel ($\vec{Q_i}$), distance metric ($dist_i$), and weight (w_i) in constructing the final summary from s_p in the following section.

B. The Encoder Layer: The encoder layer consists of 3 parallelly stacked encoder-blocks. Each encoder-block (see Fig. 3) contains an *embedding layer* and a *local-attention layer*. At every timestep t, a sentence from s_p is fed into the embedding layer of each of the three encoder-blocks. It computes a fixed-length embedding ($\vec{V_i}$) of the sentence and propagates it to the local-attention layer. Each encoder-block in our network is mapped to a unique triplet ($\vec{Q_i}$, $dist_i$, w_i) in the multiplex layer.

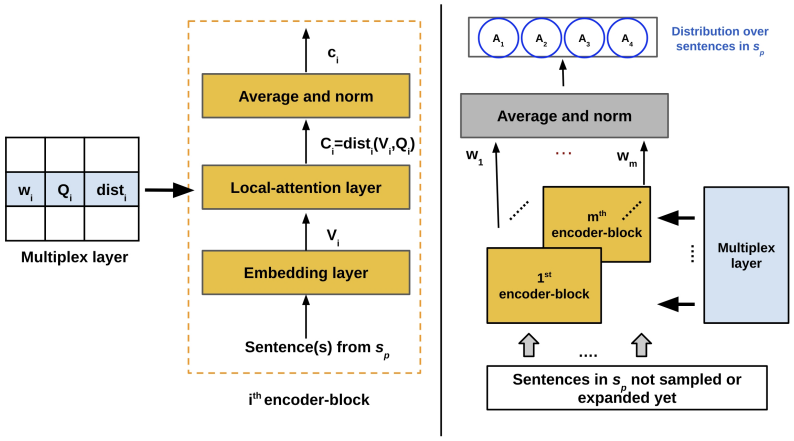

Figure 3: *The Encoder layer consists of 3 parallelly stacked encoder-blocks*

To compute local-attention (c_i) attributed to a sentence in s_p by the i^{th} encoder-block, we embed it in the same semantic space as $\vec{Q_i}$ and compute its distance from $\vec{Q_i}$ in that encoding space (Eq. 1).

$$\vec{C_{t,i}} = \frac{1}{r}\Sigma_{j=1}^{r} dist_i(\vec{V_i}, \vec{Q_i}^T[j]) \tag{1}$$

$$c_i = \frac{1}{n_i}\Sigma_{j=1}^{n_i}(\vec{C_{t,i}}[j]) \tag{2}$$

In Eq. 1, $\vec{Q_i}$ represents a kernel of dimensions $r \times n_i$ and $\vec{V_i}$ represents an embedding vector of length n_i. The embedding layer represents each sentence in s_p in the same encoding space as the kernel $\vec{Q_i}$ associated with that block. We compute the local-attention c_i by taking a column-wise average of the distance-matrix $\vec{C_{t,i}}$ (Eq. 2). The kernel $\vec{Q_i}$ is reused for all the sentences fed to the i^{th} encoder-block. The distribution $[c_1, c_2...]$ obtained this way is then normalized to derive the local-attention distribution $\vec{C_i}$ over s_p. The final attention distribution ($\vec{A}$) over s_p at timestep t is computed by normalizing the weighted average (Eq. 3) of local-attention distributions computed by each attention-head.

$$\vec{A^*} = \texttt{norm}(\frac{1}{m}\Sigma_i^m(\vec{C_i} \cdot w_i)) \tag{3}$$

It is worth noting here that attributing each encoder-block with a distinct attention-head ensures that there is a dedicated pathway to compute local attentions for every encoder-block. This allows us to parallelize the network and speed-up the decoding process when constructing the final summary.

C. The Decoder Layer: Similar to the encoder, the decoder layer also consists of 3 parallelly stacked decoder-blocks. Each decoder-block contains an *embedding layer* and a *local-attention layer*. Parameters of the i-th encoder-block and i-th decoder-block are shared. We construct a length-constrained summary s_b of the input document by processing each sentence in s_p sequentially. Depending on the remaining length-budget at each timestep, the final summary is constructed by soft-switching between a *copy* and *expand* operation. This process is guided by a sentence-level attention distribution (Eq. 3) computed over s_p. If the copy operation is selected, a sentence from s_p is copied into the final summary,

whereas the expand operation replaces a sentence with similar content from the input document in s_b. The original ordering of sentences is preserved.

The Copy Operation: The probability of copying a sentence s from the prototype summary that has not been included in the final summary (s_b) till timestep t into s_b is defined as follows: $P_c(s) = \vec{A}^t[s]$, where $\vec{A}^t$ represents a sentence-level attention distribution over s_p at timestep t. Initialized as $\vec{A}^*$ (Eq. 3), we update the attention distribution at each timestep after a copy or expand operation. If $s^* = argmax(P_c(s))$ represents the sentence copied into s_b at timestep t, we update the attention distribution by zeroing out the probability of s^* in $\vec{A}^t$ and renormalizing the resulting distribution.

The Expand Operation: If the length of our prototype summary (s_p) is less than the length-budget b, `MLS` can choose to expand a set of sentences from s_p. For each sentence $s \in s_p$, we define its *expansion-set* $E(s)$ as the sentence n-gram that is most similar to s in the input document. We determine the expansion-set $E(s)$ of a sentence s by using beam-search over all n-grams in the input document that are yet to be included in the final summary. Our search objective being maximizing $score(E) = sim(s, E) \times overlap(s, E)$. The first term in $score(E)$ denotes the average pairwise cosine similarity between s and the sentences in $E(s)$, whereas the second term denotes the fraction of tokens in s that appear in $E(s)$. To minimize across-sentence repetitions in the summary, top 4 candidates identified from the search process are re-ranked (Chen and Bansal, 2018) based on the number of repeated word bigrams and trigrams if the expansion-set is included in the final summary. We obtained best performance by initializing n with 3 and changing it to 2 at later iterations of the decoding process. If $\vec{v}_i^k$ denotes the embedding-vector of the k-th sentence in $E(s)$ computed by the embedding-layer of the i-th decoder-block, we define the probability of expanding a sentence s from the prototype summary to $E(s)$ in the final summary as follows.

$$\vec{C}^e_{i,k} = \frac{1}{r}\Sigma_{j=1}^{r}\, dist_i(\vec{v}_i^{\,k}, \vec{Q}_i^{\,T}[j]) \tag{4}$$

$$c^e_{i,k} = \frac{1}{n_i}\Sigma_{j=1}^{n_i}(\vec{C}^e_{i,k}[j]) \tag{5}$$

$$\vec{A}^e = \frac{1}{m}\Sigma_{i=1}^{m}\,(\vec{c}^e_i \cdot w_i) \tag{6}$$

In Eq. 4, $\vec{Q}_i$ denotes the semantic kernel shared between the i-th encoder-block and decoder-block. We compute the probability of including the k^{th} sentence of $E(s)$ into the final summary by computing its contribution ($c^e_{i,k}$) towards optimizing the qualitative property Φ_i encoded by $\vec{Q}_i$ first (Eq. 5). Repeating this process for all the sentences in $E(s)$, followed by normalization provides us with the distribution $\vec{c}^e_i = (c^e_{i,1}, c^e_{i,2}, ...)$. Here, $\vec{c}^e_i$ represents the probability distribution over $E(s)$. To obtain the expansion probability of a sentence in $E(s)$, we repeat this process for all 3 attention-heads and average them (Eq. 6). The probability $P_e(s)$ of expanding a sentence s from the prototype summary is obtained by averaging the expansion probability of all sentences in $E(s)$. Once a sentence s has been expanded into the final summary, we update the attention distribution by zeroing out the probability at s and renormalizing the resulting distribution.

Soft-Selection between Copy and Expansion: We define the probability $p_o(s)$ of selecting between the copy and expand operation for a sentence s in the prototype summary as follows.

$$p_o(s) = \alpha \times P_e(s) + (1 - \alpha) \times P_c(s) \tag{7}$$

$$\alpha = \begin{cases} 0 & if\ b \leq len({s_b}^*) \\ \texttt{max}(P_e(s), P_c(s)) & if\ b > len({s_b}^*) \end{cases} \tag{8}$$

In Eq. 8, ${s_b}^*$ denotes the partially constructed summary till timestep t. If the length-budget b is smaller than the length of the prototype summary s_p, the probability of including a sentence from s_p into the final summary depends on the attention distribution $\vec{A}^t$ over sentences in s_p that are not included in the final summary till timestep t. In all other scenarios, α acts as a soft-switch between copying or expanding a sentence in s_p. A sentence can be expanded only if doing so does not exceed the

length-budget. Once the probability of each sentence (and/or its expansion set) has been computed, the decoder attends to the position with the highest probability and copies/expands it into the final summary. Generation stops once $len(s_b{}^*)$ reaches b. We observed that the probability of expanding a sentence from the prototype summary (instead of copying it) increases with the allocated length-budget.

2.3 Training the Networks

We trained PG-Network and the Pointer-Magnifier network separately on a NVIDIA Titan-XP GPU with a batch size of 16. We pretrained the PG-Network on the CNN-DailyMail dataset (Nallapati et al., 2016) and then fine-tuned it on training samples of our experimental datasets. Using the evaluation script provided by (Nallapati et al., 2016), we obtained a training set of 287,226 pairs and validation set of 13,368 pairs for this dataset. All encoder-decoder weights were allowed to be updated during fine-tuning stage, following a L1-transfer (Pan and Yang, 2009) of weights from the pretrained network. The external vocabulary used in both pretraining and fine-tuning stage consisted of 80K most frequent tokens in the training samples of the CNN-DailyMail dataset, our experimental dataset or both. Learning-rate and initial accumulator values were set to 0.15 and 0.1 respectively. We used Adagrad (Duchi et al., 2011) to train the network. The encoder was fed a maximum 400 tokens and the decoder generated 100 tokens during pretraining. These values were increased to 500 and 200 respectively during fine-tuning. To prevent overfitting, we stopped training after 3000 iterations during the fine-tuning stage. With respect to the Pointer-Magnifier network, we learn the optimal values of $w_i, 1 \leq i \leq 3$ associated with each attention-head by grid-searching over the interval [-1,1] with the learning objective of maximizing ROUGE-1 score on the validation set. The optimal weights assigned to the attention-head corresponding to topic-coverage (ϕ_1) and keyword-coverage (ϕ_2) were positive, whereas information redundancy (ϕ_3) was assigned a negative weight for both of our datasets.

3 Experiments

We seek to answer three key questions in our experiments. Given a length-constrained summary s_b, (a) how similar is s_b to a gold-standard summary?, (b) is it coherent and representative of the input document? and (c) how abstractive is s_b? We answer the first two questions by evaluating the summaries generated by `MLS` over a range of success metrics on datasets belonging to two low-resource domains. We also conduct a user study to measure how representative are the summaries with respect to the input documents. A representative summary covers the main topics of the document. We answer the third question by computing the percentage of n-grams in s_b that do not appear in the input document and/or generated from the external vocabulary.

Index	Dataset	Size	Max	Median	Mean
D1	MSR Narrative	476	130	15	18.65
D2	Thinking Machines	186	82	33	33.23

Table 1: *The minimum, maximum, median, and average number of sentences in datasets D1 and D2*

A. Datasets: We evaluate `MLS` on two publicly available datasets from two low-resource domains: the MSR-Narrative (Ouyang et al., 2017) (D1) dataset and the Thinking-Machines (Brockman, 2018) (D2) dataset. The MSR-Narrative dataset contain personal stories shared by users on a social networking website. The Thinking-Machines dataset, on other hand, contains position papers on a popular scientific topic published in an educational website. Each document in both datasets is paired with a gold-standard summary. We randomly selected 25% document-pairs to construct the training set and 10% document-pairs to construct a validation set for both datasets. The rest comprised the test corpus. We present an overview of some of the important properties of both datasets in Table 1.

B. Metrics: We compare the summaries constructed by `MLS` against gold-standard summaries using METEOR (Banerjee and Lavie, 2005) and ROUGE (Lin, 2004) scores[4]. The average F_1 score of ROUGE-1, ROUGE-2 and ROUGE-L metrics obtained for both datasets are shown in Table 2. To measure the *representativeness* of a summary, we compute the average KL-divergence score between

[4]We used py-rouge (Benjamin Heinzerling, 2020) and the NLTK library to compute the ROUGE and METEOR score respectively.

Dataset	Metric	Budget = 1/32				Budget = 1/16				Budget = 1/8				Budget = 1/4				Budget = 1/2			
		MLS	A1	A2	A3	MLS	A1	A2	A3	MLS	A1	A2	A3	MLS	A1	A2	A3	MLS	A1	A2	A3
D1	ROUGE-1	**45.99**	23.44	37.46	41.65	**45.99**	30.50	37.68	43.07	**45.99**	31.27	38.05	43.50	**46.11**	41.86	43.95	44.10	**45.67**	40.67	41.13	45.50
	ROUGE-2	**35.97**	14.79	22.59	30.65	**35.97**	20.77	25.50	30.65	**35.98**	22.95	29.14	33.50	**35.60**	27.57	32.36	34.50	**36.70**	29.38	31.02	35.02
	ROUGE-L	**40.89**	21.35	32.38	37.65	**42.50**	27.9	33.07	38.92	43.01	36.25	37.62	**43.50**	**42.83**	38.83	40.95	41.07	40.18	39.60	40.74	**41.50**
	METEOR	**47.12**	18.91	24.22	45.51	**47.12**	13.07	25.02	45.60	**46.50**	20.89	30.86	43.88	**46.61**	27.26	33.05	44.65	**45.71**	27.84	32.95	45.39
D2	ROUGE-1	**40.25**	16.20	21.06	35.60	**40.0**	17.08	22.0	36.0	**40.25**	22.59	28.10	39.72	**41.01**	23.55	27.83	38.50	**44.36**	29.53	32.75	44.06
	ROUGE-2	**33.25**	11.25	17.22	26.50	**34.50**	12.0	16.75	30.05	**35.67**	14.60	19.01	31.80	**36.0**	17.90	20.06	31.0	**38.70**	20.67	23.46	36.44
	ROUGE-L	**37.17**	14.50	19.06	33.67	**37.0**	15.60	20.55	35.70	**37.05**	21.65	20.26	34.33	**37.96**	21.87	22.60	32.77	**41.50**	26.04	27.17	39.75
	METEOR	**40.22**	12.68	24.33	35.05	**44.82**	15.17	23.22	42.90	**44.82**	11.96	30.79	42.0	**42.88**	24.20	21.83	38.05	44.79	28.08	25.82	**45.70**

Table 2: *ROUGE and METEOR scores of the budgeted summaries constructed by* MLS *(highlighted column) and the baseline methods for the MSR-Narrative (D1) and Thinking Machines (D2) dataset*

the top-3 topic vectors of a summary and its input document. Following (Srinivasan et al., 2018), we measure the coherence of a summary by computing the average cosine similarity between consecutive sentences. We report the absolute difference between the coherence score computed for a summary and its input document in Table 3. We also report the KL-divergence score between sentiment vectors of a summary and the input document to check for potential biases in its polarity distribution. We used a publicly available library (Hutto and Gilbert, 2014) to derive the sentiment vectors.. Note that, lower values of $\Delta Coherence$ and KL-divergence score are desirable for a high-quality summary.

C. Baselines: We compare MLS against three baseline methods. Two of them follow a sampling based approach, while our final baseline method employs a convolutional network to construct length budgeted summaries. Our first baseline (**A1**) follows a systematic sampling based approach to construct length-controlled summaries. Initialized with a randomly selected sentence from the first k-1 sentences of the input document, it constructs the final summary by including the k-th sentence from the last sampled position. We set k = 3 in all of our experiments for both datasets. Sampling terminates when the budget limit is exceeded or the end of document is reached. Our second baseline method (**A2**) follows a weighted graph-based sampling strategy to construct budgeted summaries. It represents each sentence in the input document as a node in an undirected, complete, weighted graph. The weight assigned to an edge in this graph is equal to the pairwise cosine similarity between the connecting nodes. To construct the budgeted summary, we sample the top-K nodes of this graph using a weighted PageRank algorithm (Mihalcea and Tarau, 2004). Sampling stops when the budget is reached. Our third and final baseline method (**A3**) is a convolutional approach proposed in (Liu et al., 2018b). It is a sequence-to-sequence network with Gated Linear Units (Dauphin et al., 2017) that takes the desired length of a summary as an additional input to the initial state of the decoder network. Similar to our training protocol, we pretrain this network on the CNN-DailyMail dataset first and fine-tune it on the training samples from both of our experimental datasets. We allowed all weights to be updated during the fine-tuning phase.

3.1 Results and Discussion

We report the performance of all competing methods at five length-budgets. We specify the length-budget to construct a summary as a product of the number of tokens in the input document and a compression-budget $c \in \{\frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}\}$. Results from our experiments are presented in Tables 2 and 3. The best performance achieved for each metric is boldfaced. We highlight some of our key findings below.

3.1.1 Qualitative Evaluation at five Compression Budgets

In general, the abstractive methods (MLS and A3) outperform sampling-based approaches (see Table 2) on both datasets. MLS performs consistently well on all budgets, although performance is relatively better on smaller budgets. We obtain an absolute improvement of 4.34% and 4.65% in ROUGE-1 score & 1.61% and 5.17% in METEOR score over the convolutional baseline (A3) for datasets D1 and D2 at compression budget = $\frac{1}{32}$. At higher budgets, our performance was comparable with A3. In terms of coherence, MLS performs comparably or better than A3 (see Table 3). Smaller $\Delta Coherence$ score than A1 and A2 suggests that MLS generated more coherent summaries than these two baseline methods. Small KL-divergence between the topic distribution of a budgeted summary and input document shows that MLS generated summaries are representative of the document for both datasets. In fact, topic-coverage in summaries generated by MLS is at least 75% better than the convolutional baseline (A3) (Liu et al., 2018b), although performance becomes comparable at larger budgets as more

Dataset	Metric	Budget = 1/32				Budget = 1/16				Budget = 1/8				Budget = 1/4				Budget = 1/2			
		MLS	A1	A2	A3	MLS	A1	A2	A3	MLS	A1	A2	A3	MLS	A1	A2	A3	MLS	A1	A2	A3
D1	Topic	**0.12**	0.28	0.29	0.21	**0.12**	0.27	0.27	0.20	**0.12**	0.26	0.23	0.15	**0.13**	0.21	0.19	0.18	**0.13**	0.21	0.21	0.18
	Sentiment	**0.09**	0.22	0.19	0.11	**0.09**	0.23	0.15	0.13	**0.09**	0.19	0.15	0.12	**0.1**	0.14	0.12	**0.1**	0.16	**0.07**	0.17	0.13
	ΔCoherence	**0.08**	0.3	0.20	0.11	**0.08**	0.26	0.18	0.09	0.08	0.21	0.11	**0.07**	**0.09**	0.13	0.10	0.12	0.1	**0.06**	0.09	0.1
D2	Topic	**0.05**	0.27	0.24	0.15	**0.05**	0.27	0.25	0.16	**0.05**	0.17	0.2	0.12	**0.05**	0.08	0.08	0.11	0.03	0.03	**0.02**	0.10
	Sentiment	**0.03**	0.24	0.16	0.10	**0.03**	0.21	0.13	0.07	**0.03**	0.12	0.15	0.04	**0.03**	0.06	0.08	0.05	0.04	**0.02**	0.03	0.03
	ΔCoherence	**0.03**	0.27	0.20	0.05	**0.03**	0.18	0.12	0.10	**0.03**	0.09	0.09	0.05	**0.03**	0.05	0.05	0.06	0.04	**0.03**	**0.03**	0.04

Table 3: *Coherence and completeness of the budgeted summaries constructed by* MLS *(highlighted column) and the baseline methods for MSR-Narrative (D1) and Thinking Machines (D2) dataset*

sentences from the prototype summary are expanded to make the final summary. MLS outperfoms A1 and A2 in terms of staying true to the sentiment distribution of the input document. This can be seen from the small KL-divergence scores obtained for the sentiment distribution achieved by MLS in Table 3.

MLS generated summaries were more abstractive at higher budgets (Fig. 4). At compression budget = $\frac{1}{2}$, 27.35% tokens in the summaries constructed for dataset D1 and 8.75% tokens for dataset D2 were contributed by the external vocabulary.

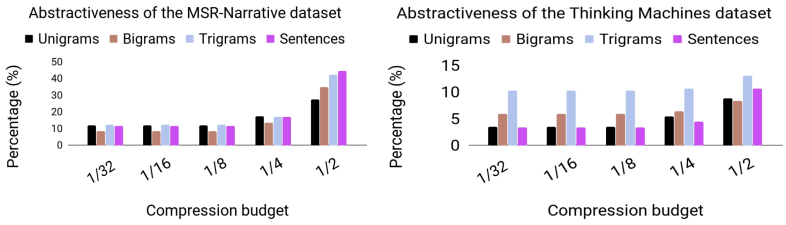

Figure 4: *Abstractiveness of* MLS *generated summaries*

3.1.2 Ablative Analysis

To investigate the effects of pretraining on end-to-end results, we compare the ROUGE-1 score of summaries constructed by MLS against an ablative baseline MLS*. It is identical to MLS except that the PG-Network was not pretrained. In our second experiment, we compare MLS against MLS+, an ablative baseline that constructs the prototype summary following a greedy heuristics (Otterbacher et al., 2006) instead of the PG-network. MLS outperforms both baselines (Fig. 5) on both datasets, thereby establishing that using PG-Network in our framework and pretraining it on the CNN-DailyMail dataset improved the quality of our final summaries. Finally, to investigate the effects of the semantic kernels introduced in the Pointer-Magnifier network, we iteratively replaced each of the three semantic kernels (Section 2.2) with a randomized kernel by shuffling its rows and columns.

We observed an absolute decrease of up to 4.30% in ROUGE-1 score and 3.75% in METEOR score for $\vec{Q}_3$, with bigger impacts in performance at higher length-budgets. Replacing $\vec{Q}_2$ with a randomized kernel, on other hand, decreased the average $\Delta Coherence$ score by approximately 45% for dataset D1 and 30% for D2 for summaries constructed at compression budget = $\frac{1}{2}$, i.e. half-length of the input document.

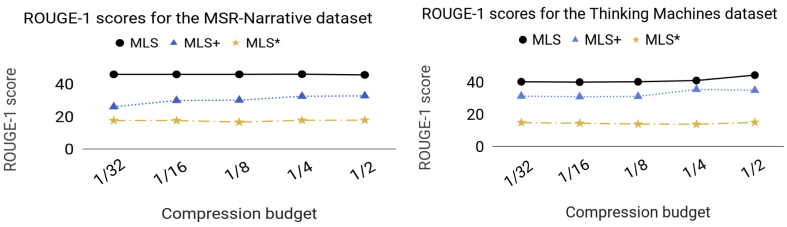

Figure 5: *ROUGE-1 score of* MLS *and the ablative baselines MLS+ and MLS* on datasets D1 and D2*

3.1.3 Human Evaluation of Length-Controlled Summaries

We conducted a study to evaluate the completeness of the summaries constructed by MLS. More specifically, we considered a scenario where the user needs to complete a fact checking task. We chose three documents from both datasets randomly and asked each participant to verify the presence of some key facts of the document in the summaries constructed by MLS and/or a baseline method. Each participant was instructed to complete the task solely based on the content of the summary and not depending on any previous knowledge. For example, the question "*Does the story tell us why the narrator was fired?*" was paired with the following summary– "*I tried to return a lost wallet to a customer who accused me*

of stealing it and then grabbed my hair. We got in a physical fight and I was fired from my job". The participants had to chose between 'Yes', 'No", and "More information required". If a participant selected the third option, a longer summary was shown with the same question. The task was terminated otherwise. In addition to MLS, A2 (the stronger extractive baseline in our experimental setup) and A3, we add two extreme settings: (a) the full-content setting in which the original document was shown, and (b) the no-content setting where no textual content (other than the question itself) was shown to a participant. The full-content setting ensured that the question could indeed be answered from the article, whereas the no-content setting ensured whether the questions contained any hint about the answer.

The task started by showing each participant a summary generated at compression budget = $1/32$. If they opted for more information to be shown, we provided a summary generated by the same method by doubling the compression budget each time until the user responded with a 'Yes' or 'No' or we reached the budget of $1/2$. The key intuition here is that if users are given a complete and representative summary, they should be able to answer the questions accurately, as a good summarization model would pick up the key concepts of the document even at shorter length-budgets, without requiring for it to be expanded further. With this in mind, we recorded task completion time and user response for each treatment. All budgeted summaries were constructed beforehand. We invited 22 graduate students to participate in the study. Each participant was shown summaries generated by at most two different methods in random order. No information on the method used was revealed to a participant at any stage. To prevent information retention, each participant was shown a summary generated from the same document only once. Using a balanced, incomplete block design (Aschbacher, 1971), each of the 10 settings (5 methods $\times$ 2 datasets) was assigned to 3 subjects. The average accuracy and task completion time recorded for each treatment is shown in Table 4. The accuracy of the no-content setting is 0 for both datasets, indicating that the questions did not contain any hint to the correct answer, whereas the full-content setting shows that overall the questions could have been answered from the original documents. When using summaries generated by MLS, the participants responded as accurately as the Full-content setting on dataset D1, while being more than two times faster, outperforming A2 and A3. For dataset D2, participants were more accurate using summaries constructed by MLS than A2. MLS performed better than A3 on one document, comparable on one and worse on one document, with an average accuracy of 0.55.

Index	Dataset	MLS	A2	A3	NC	FC
D1	**Accuracy**	**0.88**	0.55	0.55	0.0	**0.88**
	Duration (s)	**36.7**	43.69	69.08	12.0	75.6
D2	**Accuracy**	0.55	0.44	**0.66**	0.0	**0.88**
	Duration (s)	70.24	**68.9**	96.47	20.95	132.86

Table 4: *Mean accuracy and completion time using* MLS*, A2, A3, No-content (NC) and Full-content (FC) settings*

4 Conclusion

We have proposed MLS, a supervised approach to construct abstractive summaries at controllable lengths. Following an extract-then-compress paradigm, we develop the Pointer-Magnifier network – a length-aware, encoder-decoder network that constructs length-constrained summaries by shortening or expanding a prototype summary inferred from the document. The key enabler of this network is an array of semantic kernels with clearly defined human-interpretable syntactic/semantic roles in constructing the summary given a budget-length. We train our network on limited training samples from two cross-domain datasets. Experiments show that the summaries constructed by MLS are coherent and reflectively capture the main concepts of the document. Our human evaluation study also suggest the same. In the future, we would like to extend our work to construct task-driven summaries for interactive question answering tasks. Personalizing a summary based on user's past interaction model is another exciting direction of future work.

5 Acknowledgement

We would like to thank Professors Micha Elsner, Joel Bloch, Marie-Catherine de Marneffe, and Michael White for valuable discussions and comments. We would also like to thank the reviewers and folks who participated in our study for sharing critical feedback that helped improve our work.

References

Michael Aschbacher. 1971. On collineation groups of symmetric block designs. *Journal of Combinatorial Theory, Series A*.

Joris Baan, Maartje ter Hoeve, Marlies van der Wees, Anne Schuth, and Maarten de Rijke. 2019. Do transformer attention heads provide transparency in abstractive summarization? *arXiv preprint arXiv:1907.00570*.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*.

Anders Johannsen Benjamin Heinzerling. 2020. A python wrapper for the rouge summarization evaluation package. `https://pypi.org/project/pyrouge/`.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *JMLR*.

John Brockman. 2018. Thinking machines. `https://www.edge.org/annual-question/what-do-you-think-about-machines-that-think`.

Orkut Buyukkokten, Hector Garcia-Molina, and Andreas Paepcke. 2001. Seeing the whole in parts: text summarization for web browsing on handheld devices. In *The Web Conference*.

Marco Campana and Anastasios Tombros. 2009. Incremental personalised summarisation with novelty detection. In *FQAS*.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. *arXiv preprint:1805.11080*.

Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. 2017. Language modeling with gated convolutional networks. In *ICML*.

Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*.

Angela Fan, David Grangier, and Michael Auli. 2017. Controllable abstractive summarization. *arXiv preprint:1711.05217*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *ICML*. JMLR. org.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Clayton J Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *ICWSM*.

ITGP. 2017. *EU General Data Protection Regulation (GDPR).* IT Governance Limited.

Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. 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)*, pages 3543–3556.

Moniba Keymanesh, Micha Elsner, and Srinivasan Parthasarathy. 2020. Toward domain-guided controllable summarization of privacy policies. *Natural Legal Language Processing Workshop at KDD*.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. *arXiv preprint:1609.09552*.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *ICML*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. 2018a. Toward abstractive summarization using semantic representations. *arXiv preprint:1805.10399*.

Yizhu Liu, Zhiyi Luo, and Kenny Zhu. 2018b. Controlling length in abstractive summarization using a convolutional neural network. In *EMNLP*.

Chunyi Liu, Peng Wang, Jiang Xu, Zang Li, and Jieping Ye. 2019. Automatic dialogue summary generation for customer service. In *SIGKDD*.

Ahmed Magooda and Diane Litman. 2020. Abstractive summarization for low resource data using domain transfer and data synthesis. *arXiv preprint arXiv:2002.03407*.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *EMNLP*.

Milad Moradi and Nasser Ghadiri. 2018. Different approaches for identifying important concepts in probabilistic biomedical text summarization. *Artificial intelligence in medicine*, 84:101–116.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Jahna Otterbacher, Dragomir Radev, and Omer Kareem. 2006. News to go: hierarchical text summarization for mobile devices. In *SIGIR*.

Jessica Ouyang, Serina Chang, and Kathy McKeown. 2017. Crowd-sourced iterative annotation for narrative summarization corpora. In *EACL*.

Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *TKDE*.

Shantipriya Parida and Petr Motlicek. 2019. Abstract text summarization: A low resource challenge. In *EMNLP*. Association for Computational Linguistics, November.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text mining: applications and theory*.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint:1509.00685*.

Itsumi Saito, Kyosuke Nishida, Kosuke Nishida, Atsushi Otsuka, Hisako Asano, Junji Tomita, Hiroyuki Shindo, and Yuji Matsumoto. 2020. Length-controllable abstractive summarization by guiding with summary prototype. *arXiv preprint:2001.07331*.

Ritesh Sarkhel and Arnab Nandi. 2019. Visual segmentation for information extraction from heterogeneous visually rich documents. In *Proceedings of the 2019 International Conference on Management of Data*, pages 247–262.

Ritesh Sarkhel, Jacob J Socha, Austin Mount-Campbell, Susan Moffatt-Bruce, Simon Fernandez, Kashvi Patel, Arnab Nandi, and Emily S Patterson. 2018. How nurses identify hospitalized patients on their personal notes: Findings from analyzing 'brains' headers with multiple raters. In *Proceedings of the International Symposium on Human Factors and Ergonomics in Health Care*, volume 7, pages 205–209. SAGE Publications Sage CA: Los Angeles, CA.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint:1704.04368*.

Kaiqiang Song, Logan Lebanoff, Qipeng Guo, Xipeng Qiu, Xiangyang Xue, Chen Li, Dong Yu, and Fei Liu. 2020a. Joint parsing and generation for abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Kaiqiang Song, Bingqing Wang, Zhe Feng, Liu Ren, and Fei Liu. 2020b. Controlling the amount of verbatim copying in abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Balaji Vasan Srinivasan, Pranav Maneriker, Kundan Krishna, and Natwar Modani. 2018. Corpus-based content construction. In *COLING*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Christopher C Yang and Fu Lee Wang. 2003. Fractal summarization for mobile devices to access large documents on the web. In *The Web Conference*.