A Proposal: Interactively Learning to Summarise Timelines by Reinforcement Learning

Yuxuan Ye
Intelligent Systems Laboratory
University of Bristol
yuxuan.ye@bristol.ac.uk

Edwin Simpson
Intelligent Systems Laboratory
University of Bristol
edwin.simpson@bristol.ac.uk

Abstract

Timeline Summarisation (TLS) aims to generate a concise, time-ordered list of events described in sources such as news articles. However, current systems do not provide an adequate way to adapt to new domains nor to focus on the aspects of interest to a particular user. Therefore, we propose a method for interactively learning abstractive TLS using Reinforcement Learning (RL). We define a compound reward function and use RL to fine-tune an abstractive Multi-document Summarisation (MDS) model, which avoids the need to train using reference summaries. One of the sub-reward functions will be learned interactively from user feedback to ensure the consistency between users' demands and the generated timeline. The other sub-reward functions contribute to topical coherence and linguistic fluency. We plan experiments to evaluate whether our approach could generate accurate and precise timelines tailored for each user.

1 Introduction

Notable events often happen over a long period. For example, COVID-19 caused immeasurable damage around the world, lasting for more than a year. When reviewing different aspects of the disaster, the huge number of reports and news articles makes it difficult to trace the development of events such as outbreaks, policy interventions and vaccination efforts. TLS can solve this problem by identifying significant dates and summarising events of sub-topics.

Most prior TLS works focused on producing *extractive* timelines, which copies the original sentences from input documents (Martschat and Markert, 2018; Nguyen et al., 2014; Yan et al., 2011). Irrelevant and repeated information may be extracted in this process, decreasing the quality of the generated timelines. Abstractive timeline summarisation methods can address this problem (Steen and Markert, 2019; Barros et al., 2019) but few neural network models have been proposed due to the lack of reference timelines for supervised learning. Producing reference timelines by human requires expertise to capture important temporal information and sub-events from the source documents, thus it is extremely expensive. In MDS tasks, researchers have tried heuristics-based and unsupervised methods to address the reference data shortage problem (Ryang and Abekawa, 2012; Rioux et al., 2014). However, their results on some evaluation metrics, like ROUGE-2, only reached half of the upper bound. Gao et al. (2018) showed that interactive learning could improve the performance of an MDS system via leveraging users' preference, which is relatively easy to obtain, and does not require reference summaries. Therefore, we take inspiration from their work to propose an interaction-based abstractive TLS framework.

Martschat and Markert (2018) treated the TLS task as an MDS task and proposed a modular summarisation method, which achieved the state of the art and is adaptable. However, its adaptation requires abstracting mathematical constraints from concrete requirements. This contrasts with interactive learning (IL), which greatly decreases the cognitive burden for humans by receiving user feedback to refine summaries (Gao et al., 2018; Lin et al., 2010). Comparing to traditional approaches, interaction enables the model to learn from the users, thus it is possible to accurately tailor and refine timeline summaries according to users' demands.

In this paper, we propose an interaction-based abstractive timeline summarisation framework using deep RL. By learning a reward signal from user feedback, we can fine-tune a pretrained MDS model for the TLS task via a small number of interactive learning rounds. Therefore, our frame-

Proceedings of the First Workshop on Interactive Learning for Natural Language Processing , pages 25–31
August 5, 2021. ©2021 Association for Computational Linguistics

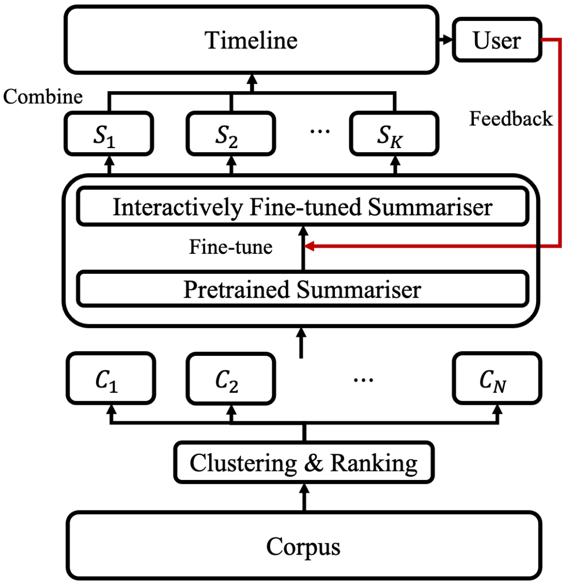

Figure 1: The workflow of our event detection timeline summarisation method

work should be capable of generating timeline summaries with high text quality after enough episodes of training. And we plan both simulation and real-user experiments to evaluate the framework on two benchmark TLS datasets, Timeline17 (Binh Tran et al., 2013; Tran et al., 2013) and Crisis (Tran et al., 2015).

The workflow of our model (Figure 1) mainly follows the event detection method, CLUST (Ghalandari and Ifrim, 2020), which identifies sub-events first and then generates summaries for them. Due to the RL-based interactive learning process in the framework, our model can be automatically adapted to new topics and adjusted by users' interests.

1. Firstly, we embed source documents into vectors and cluster them in vector space. Each cluster represents a sub-event in a large topic;

2. In the next step, we assign a date to each cluster. And they will be ranked by a metric to identify important sub-events;

3. Then it comes to our RL-based interactive learning process.

    (a) An abstractive MDS model will generate summaries for each sub-event. All summaries will be ordered by date to form a timeline.

    (b) The user can preview the timeline in this step and respond by expressing preferences over keywords or by comparing the new summary to an earlier version.

    (c) Using a reward function that evaluates the consistency between the produced timeline and those user preferences, offline RL then tunes the model and starts another round of interactive learning.

Our main contribution is a proposed interactive method for generating timelines for news, which adapts to user feedback through RL fine-tuning.

2 Related Work

Extractive Timeline Summarisation Prior extractive methods (Martschat and Markert, 2018; Ghalandari, 2017) defined several objective functions to assess the quality of timelines, including coverage of summaries and temporal information. These methods greedily select one sentence in each iteration to maximise the combined objective function. Our reward function is also modular but lacks monotonicity and submodularity, hence we use RL instead of a greedy algorithm.

Interactive Summarisation Instead of producing reference texts by crowdsourcing, obtaining information (e.g., keywords) via user interaction can be more practical to obtain training data. Liu et al. (2012) outperformed previous extractive MDS approaches on ROUGE-based metrics by querying topic words from users. Gao et al. (2018) collected pairwise comparisons between summaries from simulated users, which are then used to train a ranker without any reference data, and fixed the efficiency issue of IL. Due to the similarity between the MDS and TLS task, IL is expected to solve the reference timelines shortage problem as well, without increasing many computation expenses. So we introduce interaction into an RL-based TLS model for the first time.

Reinforcement Learning in Natural Language Generation (NLG) Recent research on applying RL on NLG tasks has received some success. Some prior works on dialogue systems (Song et al., 2020; Mesgar et al., 2020) utilized RL-based fine-tuning method to ensure the factual consistency of the response. In automatic summarisation (Gao et al., 2018, 2019; Simpson et al., 2020), IL is applied to learn a reward function from users, so that RL agents could learn a policy to summarise text indirectly under users' guidance. However, for the

TLS task, we are the first to use RL to generate summaries for key dates.

3 Method

All components of our method shown in Figure 1 will be introduced below.

3.1 Event Detection Timeline Summarisation

Clustering For each input document, we use the sentence-transformer (Reimers and Gurevych, 2019) based on DistilRoBERTa (Liu et al., 2019) to embed its sentences. Then we represent the document by the mean of the sentence vectors expecting that dense vectors could capture more information in text than TF-IDF vectors, as used in Steen and Markert (2019) and Ghalandari and Ifrim (2020).

Next, we use Affinity Propagation (AP) (Frey and Dueck, 2007) to cluster all the documents. AP is an unsupervised method, which automatically determines the number of clusters. AP uses an affinity matrix A, constructed by the Euclidean distance of each pair of document vectors.

To detect events accurately, we add constraints to the clustering algorithm. If two reports were published too apart from each other, although, with a small distance in vector space, they should be considered to belong to two similar but different sub-topics. In our model, we keep the setting of prior work (Steen and Markert, 2019). If d_i and d_j were published no more than t day(s) apart, $A_{i,j} = -\|\vec{d_i} - \vec{d_j}\|_2^{1/2}$, otherwise it will be assigned by 0.

Date Assignment By clustering all the documents, reports describing the same event are gathered. However, temporal information is equally as important as summaries in TLS, which differs from MDS. Martschat and Markert (2018) and Chen et al. (2019) adapted MDS methods to make them temporally sensitive. Both received outstanding results. In our work, we use HeidelTime (Strötgen and Gertz, 2015) to identify and count date expressions in documents. Following Ghalandari and Ifrim (2020), we assign each cluster with the most frequently mentioned date in it.

Cluster Ranking Some clusters contain less important information than others. According to Ghalandari and Ifrim (2020), the importance of a cluster is in proportion to the number of sentences that mentions the assigned date to some extent. To capture useful information, we use the same setting and only summarise the top-k important clusters.

Cluster Summarisation & Timeline Construction Summarising the sub-topic of a key date can be regarded as an MDS task, as each event has multiple sources. We plan to fine-tune an abstractive MDS model for this task, which will be introduced later. After all the top-k clusters are summarised, we combine all the summaries by date to generate a timeline. We follow the setting of Ghalandari and Ifrim (2020), which skips a cluster when its date is already used by another prior cluster.

3.2 Interaction

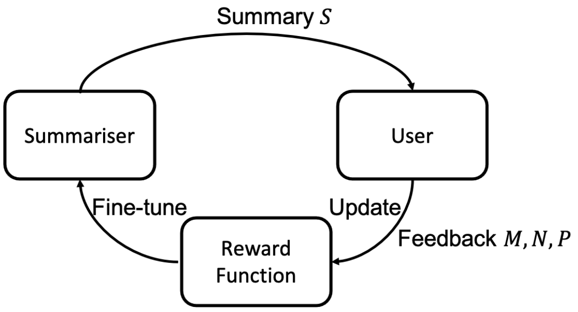

Figure 2: A view of interaction process

Every time the timeline is generated, the user can preview it and provide several types of feedback such as keywords and dates that must be included or excluded, and expressing preferences against previous version of the timeline. Given these feedback, we can renew our reward function and fine-tune the summariser via hundreds of RL episodes. Then we can produce a new timeline to start another round of interactive learning. After several interactive learning rounds, our model would be able to generate and tailor a high-quality timeline for the user.

3.3 RL-based fine-tuning

Timeline17		
	AR-F1	AR-F2
CLUST	0.082	0.02
PEGASUS-Multi_News	**0.089**	0.019

Table 1: Performance of two methods evaluated by Alignment ROUGE-1 and Alignment ROUGE-2.

PEGASUS We use PEGASUS (Zhang et al., 2020) to solve the MDS task on each cluster. PEGASUS is an abstractive summariser providing various fine-tuned versions. PEGASUS-Multi_News is fine-tuned on Multi-News (Fabbri et al., 2019) to summarise news articles. We found that PEGASUS-Multi_News outperforms the state-of-the-art extractive event detection method, CLUST (Ghalandari and Ifrim, 2020), when applying it directly on clusters without fine-tuning (Table 1). Therefore, it provides a strong basis for our following work.

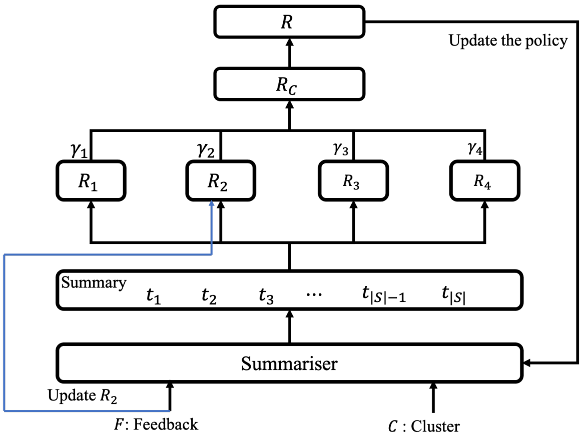

Figure 3: A view of our RL method

PEGASUS-RL Although PEGASUS is powerful enough to generate high-quality summaries, we still need RL to ensure the summaries are topically coherent and linguistically fluent. The PEGASUS model generates summaries token-by-token. When the last token, i.e. ⟨eos⟩, is generated, the reward component will assess the quality of the summary and produce a reward signal to update the summarising policy (Figure 3). This whole process will tune the parameters of PEGASUS so that it enhances the quality of the generated summary as well.

Action and Reward Function Let $D = (d_1, d_2, \ldots, d_{|D|})$ be a document cluster describing the same sub-topic. $P = (p_1, p_2, \ldots, p_{|P|})$ denotes the preferences between different versions of the generated timelines. Assuming that $p_1, p_2, \ldots, p_{|P|}$ are several different pairwise labels, collected over a number of rounds, comparing several different versions of the timeline. The words, dates and keyphrases that the user wants to include and exclude are marked as $M = (m_1, m_2, \ldots, m_{|M|})$ and $N = (n_1, n_2, \ldots, n_{|N|})$ individually. And $S = (t_1, t_2, \ldots, t_{|S|})$ is the summary generated for cluster D. Our goal is to fine-tune a single model to generate a summary S, for each cluster D that is linguistically fluent and topically coherent with any d_i and consistent with any piece of feedback p_i, m_i, n_i.

We regard each token generation process in Figure 3 as an action of PEGASUS. Our model is expected to generate a summary with topical coherence, linguistic fluency and consistency with the user's demands for each cluster. Thus, a compound reward function is proposed, which consists of four sub-reward functions: R_1 guarantees topical coherence with the cluster, R_2 enforces consistency with each piece of individual user feedback, R_3 and R_4 contribute to the linguistic fluency of the produced summaries. The reward of the cluster D is the weighted sum of them.

$$R_C = \gamma_1 R_1 + \gamma_2 R_2 + \gamma_2 R_3 + \gamma_4 R_4 \tag{1}$$

where $\gamma_{1,2,3,4}$ are the normalization factors that sum to one. The whole training signal R is the sum of k selected clusters' rewards.

Topical coherence sub-reward (R_1 and R_2) Topical coherence is the pivotal property of a summary. We measure how topically coherent the summary S is with a cluster D by their cosine similarity.

$$R_1 = \cos(\vec{S}, \vec{D}) \tag{2}$$

R_2 is the core reward function in the fine-tuning process, which will be updated in each interactive learning round. We embed all the keywords in M and N to dense vectors and measure their topic coherence by cosine similarities. Due to N represents the words that the user wants to exclude, we set its reward to be negative. To accommodate pairwise preference labels, we learn a ranking function using a random utility model (Thurstone, 1927; Mosteller, 2006). This provides a scoring function that should also be added to R_2.

$$\begin{aligned} R_2 &= w_1 score(\vec{S}, \vec{P}) \\ &+ w_2 \sum_{m_i \in M} \cos(\vec{S}, \vec{m_i}) \\ &- w_3 \sum_{n_i \in N} \cos(\vec{S}, \vec{n_i}) \end{aligned} \tag{3}$$

where $w_{1,2,3}$ are the normalization factors.

Linguistic fluency sub-reward (R_3 and R_4) Prior work (Mesgar et al., 2020) has shown that applying RL to improve evaluation metrics' results might lead to decreasing in linguistic quality. To avoid that, we apply two sub-reward functions to our model. R_3 utilizes a language model which has been fine-tuned on a similar news dataset:

$$R_3 = \frac{\alpha - N(S)}{\alpha} \tag{4}$$

where $N(\cdot)$ is the Negative Log-likelihood loss function, and α is the maximum of $N(\cdot)$ so that it can normalize R_3. R_4 reduces repeated words in summaries, by penalizing repeated unigrams:

$$R_4 = 1 - \frac{\#repeated\_tokens\_in\_summary}{\#tokens\_in\_summary} \tag{5}$$

Training In this work, RL attempts to learn a policy P_θ that generates a summary maximizing the expectation of the reward function.

$$L = E_{S \sim P_\theta}[R(S, (C, F))] \tag{6}$$

However, RL is known for high variance issue when sampling the gradient. To solve this problem, we plan to run several hundred episodes of RL to increase the size of the sample and reduce the variance.

In addition, according to Mnih et al. (2016) and Mesgar et al. (2020), we can tune the policy function by actor-critic, which could further reduce variance in learning. In actor-critic algorithm, the policy function P_θ is regarded as the actor, and we define the residual of temporal difference Ψ_t to be the critic. Although Ψ_t is a biased estimation of the reward function R, we can reduce the variance via replacing the reward function R in the policy gradient equation (7) by Ψ_t, as in the following:

$$g = E\left[\sum_{t=0} \Psi_t \nabla_\theta \log P_\theta(a_t|s_t)\right] \tag{7}$$

4 Plan for Evaluation

As a kind of summarisation task, correctly extracting temporal information is the special challenge of TLS, which makes the evaluation more complex as well. In our work, we plan to evaluate our model by the suitable evaluation metrics proposed by Martschat and Markert (2017).

Concatenation ROUGE Discard all dates and concatenate all summaries in the reference and the output timeline. Evaluate ROUGE on two concatenated texts.

Alignment ROUGE Align the output timeline with the reference by the similarity and distance of their dates and apply ROUGE on them. Aligned summaries with distant dates will be penalized.

User feedback will be generated through mixed simulations, as in Gao et al. (2019) and studies with real users. Simulations will rely on references, from which keywords and dates can be extracted. Pairwise preferences can be simulated by comparing two summaries to a reference using ROUGE and selecting the highest-scoring summary. The system will be tested with different feedback types (keywords, dates, inclusion/exclusion, and preferences) to determine whether these forms of interaction are feasible to improve the summaries. However, the simulated user labels will be noisy, so we intend to evaluate with real users once we have developed a working system.

5 Summary

We propose an interactive method to summarise timelines without reference data. In each interactive learning round, we first update the reward function, and then use RL to fine-tune a huge neural network model. Then the model will generate summaries for each of the important sub-events, which are identified by textual similarity to the articles in the corpus. All the summaries will be ordered by their assigned dates to form a timeline. The user can preview the timeline and give feedback to start another round of interactive learning. Part of our method has been implemented, including PEGASUS to summarise event clusters but without RL or user feedback. Given the current experiment results, we can expect better performance after the interaction part implemented. The challenge remains in RL and designing suitable modes of interaction. We will move forward to our planned experiments and report our results in future work.

References

Cristina Barros, Elena Lloret, Estela Saquete, and Borja Navarro-Colorado. 2019. Natsum: Narrative abstractive summarization through cross-document timeline generation. *Information Processing & Management*, 56(5):1775–1793.

Giang Binh Tran, Mohammad Alrifai, and Dat Quoc Nguyen. 2013. Predicting relevant news events for timeline summaries. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 91–92.

Xiuying Chen, Zhangming Chan, Shen Gao, Meng-Hsuan Yu, Dongyan Zhao, and Rui Yan. 2019. Learning towards abstractive timeline summarization. In *IJCAI*, pages 4939–4945.

Alexander R Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. *arXiv preprint arXiv:1906.01749*.

Brendan J Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. *science*, 315(5814):972–976.

Yang Gao, Christian M Meyer, and Iryna Gurevych. 2018. April: Interactively learning to summarise by combining active preference learning and reinforcement learning. *arXiv preprint arXiv:1808.09658*.

Yang Gao, Christian M Meyer, Mohsen Mesgar, and Iryna Gurevych. 2019. Reward learning for efficient reinforcement learning in extractive document summarisation. *arXiv preprint arXiv:1907.12894*.

Demian Gholipour Ghalandari. 2017. Revisiting the centroid-based method: A strong baseline for multi-document summarization. *arXiv preprint arXiv:1708.07690*.

Demian Gholipour Ghalandari and Georgiana Ifrim. 2020. Examining the state-of-the-art in news timeline summarization. *arXiv preprint arXiv:2005.10107*.

Jimmy Lin, Nitin Madnani, and Bonnie Dorr. 2010. Putting the user in the loop: interactive maximal marginal relevance for query-focused summarization. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 305–308.

Yan Liu, Sheng-hua Zhong, and Wenjie Li. 2012. Query-oriented multi-document summarization via unsupervised deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Sebastian Martschat and Katja Markert. 2017. Improving rouge for timeline summarization. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 285–290.

Sebastian Martschat and Katja Markert. 2018. A temporally sensitive submodularity framework for timeline summarization. *arXiv preprint arXiv:1810.07949*.

Mohsen Mesgar, Edwin Simpson, Yue Wang, and Iryna Gurevych. 2020. Generating persona-consistent dialogue responses using deep reinforcement learning. *arXiv preprint arXiv:2005.00036*.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR.

Frederick Mosteller. 2006. Remarks on the method of paired comparisons: I. the least squares solution assuming equal standard deviations and equal correlations. In *Selected Papers of Frederick Mosteller*, pages 157–162. Springer.

Kiem-Hieu Nguyen, Xavier Tannier, and Véronique Moriceau. 2014. Ranking multidocument event descriptions for building thematic timelines. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1208–1217.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Cody Rioux, Sadid A Hasan, and Yllias Chali. 2014. Fear the reaper: A system for automatic multi-document summarization with reinforcement learning. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 681–690.

Seonggi Ryang and Takeshi Abekawa. 2012. Framework of automatic text summarization using reinforcement learning. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 256–265.

Edwin Simpson, Yang Gao, and Iryna Gurevych. 2020. Interactive text ranking with bayesian optimization: A case study on community qa and summarization. *Transactions of the Association for Computational Linguistics*, 8:759–775.

Haoyu Song, Wei-Nan Zhang, Jingwen Hu, and Ting Liu. 2020. Generating persona consistent dialogues by exploiting natural language inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8878–8885.

Julius Steen and Katja Markert. 2019. Abstractive timeline summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 21–31.

Jannik Strötgen and Michael Gertz. 2015. A baseline temporal tagger for all languages. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 541–547.

Louis L Thurstone. 1927. A law of comparative judgment. *Psychological review*, 34(4):273.

Giang Tran, Mohammad Alrifai, and Eelco Herder. 2015. Timeline summarization from relevant headlines. In *European Conference on Information Retrieval*, pages 245–256. Springer.

Giang Binh Tran, Tuan A Tran, Nam-Khanh Tran, Mohammad Alrifai, and Nattiya Kanhabua. 2013. Leveraging learning to rank in an optimization framework for timeline summarization. In *SIGIR 2013 Workshop on Time-aware Information Access (TAIA*.

Rui Yan, Xiaojun Wan, Jahna Otterbacher, Liang Kong, Xiaoming Li, and Yan Zhang. 2011. Evolutionary timeline summarization: a balanced optimization framework via iterative substitution. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 745–754.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.