Optimizing Multidocument Summarization by Blending Reinforcement Learning Policies

DiJia Su, Difei Su, John M. Mulvey, and H.Vincent Poor, *Life Fellow, IEEE*

***Abstract*—We consider extractive summarization within a cluster of related texts (multidocument summarization). Unlike single-document summarization, redundancy is particularly important because sentences across related documents might convey overlapping information. Thus, sentence extraction in such a setting is difficult because one will need to determine which pieces of information are relevant while avoiding unnecessary repetitiveness. To solve this difficult problem, we propose a novel reinforcement learning-based method Policy Blending with maximal marginal relevance and Reinforcement Learning (*PoBRL*) for solving multidocument summarization. PoBRL jointly optimizes over the following objectives necessary for a high-quality summary: importance, relevance, and length. Our strategy decouples this multiobjective optimization into different subproblems that can be solved individually by reinforcement learning. Utilizing PoBRL, we then blend each learned policies to produce a summary that is a concise and a complete representation of the original input. Our empirical analysis shows high performance on several multidocument datasets. Human evaluation also shows that our method produces high-quality output.**

***Impact Statement*—Automatic multidocument summarization has a significant technological impact on society. It can be used as a tool to reduce the time that we spent on analyzing and reading articles. Providing high-quality and comprehensive summarization helps users to efficiently navigate through useful data and search for areas of interest. This has become much more important as we have entered the era of big data. In addition, multidocument summarization offers concise and accurate reports across multiple relevant information sources. It reinforces the credibility of information and demonstrates more objectivity over human manual summarization. In this article, we develop a method for multidocument summarization providing a concise and complete representation of the original input. Given that our novel methodology fuses multiple overlapping information sources, in the future, such an application can be utilized to filter, prioritize, and rank news and data according to a user's preferences and needs.**

***Index Terms*—Artificial intelligence, deep learning, deep reinforcement learning, document summarization, machine learning, natural language processing (NLP), reinforcement learning (RL).**

Manuscript received 25 October 2021; revised 9 April 2022 and 4 June 2022; accepted 12 August 2022. Date of publication 26 August 2022; date of current version 24 May 2023. This work was supported in part by the U.S. National Science Foundation under Grant CCF-1908308. This article was recommended for publication by Associate Editor Q. Li upon evaluation of the reviewers' comments. *(Corresponding author: DiJia Su.)*

DiJia Su and H.Vincent Poor are with the Department of Electrical and Computer Engineering, Princeton University, Princeton, NJ 08540 USA (e-mail: dsu@princeton.edu; poor@princeton.edu).

Difei Su is with the Department of Computer Science, University of British Columbia, Vancouver, BC V6Z1T8, Canada (e-mail: difei.su@alumni.ubc.ca).

John M. Mulvey is with the Department of Operation Research and Financial Engineering, Princeton University, Princeton, NJ 08540 USA (e-mail: mulvey@princeton.edu).

Digital Object Identifier 10.1109/TAI.2022.3201807

I. Introduction

SUMMARIZATION is the process of creating a concise and comprehensive representation from a large and complex information input. It has important applications in information retrieval and natural language processing (NLP) domains. Traditionally, summarization can be broadly categorized as extractive or abstractive summarization [1]. In extractive text summarization, salient information and key sentences are extracted directly from the original text without modification. In abstractive text summarization, summary is built by paraphrasing sentences or generating new words not in the original text.

In this research article, we examine extractive summarization for multidocument inputs.[1] One of the key challenges is redundancy since sentences from adjacent documents may transmit redundant information. To solve this problem effectively, one must assess which information is useful while avoiding excessive repetition. In this sense, finding an optimal summary can be viewed as a combinatorial optimization problem. One way to solve this problem is the *maximal marginal relevance* (MMR) algorithm [2]. In the MMR approach, each sentence is extracted by maximizing its relevance while minimizing over redundancy. Since MMR is an iterative and greedy algorithm, it only achieves local optimality. As another approach, in McDonald [3] developed a globally optimal MMR method by reformulating it as an inference problem and solving it using integer linear programming.

In this article, we present a novel approach to tackle multidocument summarization. We formulate this as a multiobjective optimization problem, and we seek to jointly optimize over the following three key metrics necessary for a high-quality output [3].

1) *Importance:* Only the relevant and critical information should be extracted.
2) *Redundancy:* The extracted information should not be self-repetitive.
3) *Length:* The final output should be as concise as possible.

Our approach optimizes these three objectives simultaneously and is able to produce output that is a condensed and complete summary of the original content. At a high level, our method utilizes MMR to navigate through a complicated and overlapping multidocument space. We present ourPolicy Blending with maximal marginal relevance and Reinforcement Learning (PoBRL) method to decouple this multiobjective optimization problem into smaller problems that can be solved using reinforcement

[1]Note: multidocument here means multiple documents about the same topic.

learning (RL). Then, through our PoBRL method, we present a policy blending method that integrates the learned policies together so that the combined policy is of high quality in the context of the three identified objectives. Through our newly proposed model and empirical evaluation on the Multi-News [4] and DUC-04 datasets, our approach demonstrates strong results. Our *contributions* are given as follows:

1) We propose a novel PoBRL algorithmic method that jointly optimizes over the three identified metrics by objective decoupling, independent policy learning. We combine the learned policies by "blending."
2) We propose a novel, simple, and effective multidocument summarization model that leverages the hierarchical structure of the multidocument input.
3) Our empirical performance on multidocument datasets shows high performance. Human evaluation also suggests that our method produces fluent and high-quality output.

Algorithm 1: Greedy Iterative MMR Algorithm.

S$\leftarrow \{\}$
While len $<$ maxLen **do**
  for s_i in $R \backslash S$ **do**
    score$_{imp}$ = Importance(s_i)
    for s_j in S **do**
      score$_{red}$ = max(score$_{red}$, Redundancy(s_i, s_j))
    end for
    if $MMR - Score < \lambda$ score$_{imp} - (1-\lambda)$score$_{red}$ **then**
      $s_{\text{pick}} = s_i$
      Update MMR-Score
    end If
  end for
  $S \leftarrow S \cup \{s_{\text{pick}}\}$
end while

II. Related Work

Extractive multidocument summarization can be viewed as a discrete optimization problem. In this setting, each source sentence can be regarded as a string of binary variables. Under a predefined summary length as the constraint, the optimization problem needs to decide which sentences should be included in the final summary. Several techniques have been introduced to solve this problem. For instance, Lin and Bilmes [5] proposed to solve it by maximizing submodular functions under a budget constraint. McDonald [3] and Gillick and Favre [6] reformulated the problem and solved it with integer linear programming.

With the recent advances in machine learning, deep learning techniques have gained traction in summarization. For instance, a more recent approach combines BERT with text-matching for extractive summarization [7]. Lebanoff et al. [8] and Zhang et al. [9] adapted a network trained on single document corpus directly to multidocument setting. Jin et al. [10] proposed a multigranularity network for multidocument summarization. In [11], this problem was solved using a graph convolutional network. Zhong et al. [7] proposed a multitask learning approach. Other related works include [12], [13], [14], [15], [16], [17], [18], and [19].

Alternatively, there are RL approaches to solving summarization. Narayan et al. [20] utilized REINFORCE [21] (an RL algorithm) to train the model for extractive single-document summarization. A similar approach was also taken in [22] by including a question-focus reward. Chen and Bansal [23] performed abstractive summarization following sentence extraction. In [24], REINFORCE was being applied for abstractive summarization. A more stable strategy (actor–critic) has been considered in [25]. A similar approach has also been investigated in [26] and [27]. However, all these approaches focus on training a *single* RL policy for solving the summarization problem.

To the best of our knowledge, our PoBRL approach that blends *two* RL policies (with one policy optimizing for importance, and another policy optimizing for redundancy) for solving the multidocument summarization problem has not been explored in the NLP literature. The closest related work is [28], in which the authors proposed a *curriculum learning-based* RL method for compressing (mixing) multiple RL policies to solve a high-dimensional control task. Different from their curriculum learning approach, here we blend (mix) different policies by simply taking the MMR combination of them as in (1) [and (12)], a much simpler and intuitive approach for solving an *NLP* task (multidocument summarization) without introducing an extra layer of complexity.

III. Background

A. Maximal Marginal Relevance

We leverage MMR to navigate the complicated and overlapping sentence space in the multidocument. MMR provides a balance between relevance and diversity. Formally, MMR is defined as

$$\begin{aligned}\text{MMR} = \max_{s_i \in R \backslash S}(&\lambda \text{Importance}(s_i) \\ &- (1-\lambda)\max_{s_j \in S}\text{Redundancy}(s_i, s_j)) \end{aligned} \quad (1)$$

where S is the produced summary, s_i is the ith sentence from the documents R excluding S, and λ is a parameter balancing between importance and diversity. In general, importance and redundancy functions can be in any form. In practice, (1) is implemented as an iterative and greedy algorithm that loops through each sentence in the documents for a predefined summary length.

Accordingly, the MMR from (1) is implemented as an iterative and greedy algorithm. The complete procedure is shown in Algorithm 1.

B. Motivation for RL

As noted in [3] and [5], such an iterative and greedy approach is nonoptimal because the decision made at each time step is local and does not consider its downstream consequences. The sequential nature of the algorithm naturally fits within the RL framework in which each sentence extraction action is optimized for future (downstream) cumulative expected reward.

Formally, we view the sentence extraction as a Markov decision process. We define a state $X_t = \{R, S\}$, which is a combination of input documents and extracted set of sentences at time t respectively. At each time step, our policy makes a decision a_t determining which sentence s_t is to be extracted. We define our policy π that *searches* over possible sentences across the space. Instead of having a predetermined length of summary, we let our policy π to optimize it directly. We define actions of two types

$$a_t = \begin{cases} s_t \sim P_\pi(.|X_t) \\ \text{STOP extraction.} \end{cases}$$

To implement this in practice, we add the "STOP" action as an additional action onto the action set (the number of sentences in the input). Each action in our sentence extraction setup corresponds to a sentence extraction probability. To give an example, if the input text contains ten sentences, then we will have ten actions, and the policy π will learn a probability distribution over these ten sentences. To incorporate the stop "action" into this framework, we include one additional action. So the resulting action set is 10 (sentences) + 1 (stop) = 11 actions. When the last action (the stop) is chosen, we terminate the entire extraction process. With this action setup, our extractive agent can design the optimal strategy as being when to stop extraction. When it instead decides to continue extracting sentences, each sentence in the input is extracted according to the probability $P_\pi(\cdot|X_t)$, where X_t is the state which is used to keep track of what has been extracted so far (S), and what has not be extracted (R). Our goal is to derive an optimal extraction policy $\pi^* = \text{argmax}_{\pi\in\Pi} E(\sum_t r_t|X_t, a_t)$, where r_t is the reward contribution of adding sentence s_t onto the system summary S.

Let $S = \{s_i|i \le m\}$ be a set of m sentences that constitute a system summary, and let $\rho(S)$ be the ROUGE-N score of S, where ROUGE-N evaluates the n-gram overlaps between the gold summary and the system summary S. Then,

$$\rho(S) = \frac{1}{R_N} \sum_{g\in S^*} (\min(F_s(g), F_s^*(g))) \tag{2}$$

where S^* is the gold summary, $F_s(g)$ denotes the number of times that the n-gram of g type occurs over S, and R_N denotes the number of n-gram tokens.

Let $C_{Y,S^*} = \min(F_Y(g), F_{S^*}(g))$ equal the contribution of the n-gram g, and $\epsilon(a \wedge b) = \sum_{g\in S^*} \max(C_{a,S^*}(g) + C_{b,S^*}(g) - F_{s^*}(g), 0)$ be the redundancy between a, b in the summary. Then, following [29], we can write

$$\begin{aligned} \rho(S) &= \sum_{i=1}^{m} \rho(s_i) \\ &\quad + \sum_{k=2}^{m} (-1)^{k+1} \left(\sum_{1\le i_1\le \ldots \le i_k \le m} \epsilon^{(k)}(s_i \wedge .. \wedge s_{i_k}) \right) \\ &\approx \sum_{i}^{m} \rho(s_i) - \sum_{a,b\in S, a\neq b} \tilde{\epsilon}(a \wedge b) \\ &\approx \sum_{i}^{m} \text{imp}(s_i) - \sum_{a,b\in S, a\neq b} \text{red}(a, b) \end{aligned} \tag{3}$$

where $\tilde{\epsilon}(\cdot)$ approximates the redundancy, $\text{imp}(\cdot)$ denotes the importance function, and $\text{red}(\cdot)$ denotes the redundancy function. In other words, to maximize the ROUGE score, we can optimize the importance of each sentence while subtracting (minimizing) over the redundancy.

Now, we view the sentence extraction process by defining a policy π that *searches* for the optimal set of sentences in the input text space. We are looking for the set of sentences (final system summary) that is the concise, nonrepetitive, and complete representation of the original input. In the multidocument setting, however, related documents might contain large numbers of overlapping sentences. As shown in Fig. 1, for each ground truth sentence (labeled as blue), there might exist more than one candidate sentence coming from multiple related text. For the optimal summary S^*, we seek to extract the single best sentence (colored dark orange) from a pool of similar looking sentences (denoted as "relevant sentence space" in the figure). The optimal sentence selection strategy contains two policies, where the first search policy π^{imp} learns to discover the most relevant sentence space. From this smaller space, we have the second policy π^{red} learning to search for the most relevant but nonredundant sentence (denoted as the most optimal sentence).

For effective navigation in such a search space, we propose our PoBRL method (Algorithm 2). Based on (3), our method decomposes the importance and redundancy objectives independently into two subproblems. Since each problem has its own local objectives, to maximize (3), we design policies π^{imp} and π^{red} to optimize for each objective independently. At a high level, the π^{imp} helps the policy to narrow down the search by identifying the most relevant (or important) regions. From these regions, π^{red} searches for the most relevant and diverse (nonredundant) sentence. This search procedure is graphically illustrated in Fig. 1, where π^{imp} leads the search toward the region containing candidates (1,2,3) which are sentences relevant to the ground-truth sentence; then, in this region, π^{red} leads the search to discover the most relevant and non-redundant sentence (denoted as candidate 3*). Connecting π^{red} and π^{imp} together formulates the optimal policy π^{PoBRL}, which selects the optimal sentence.

C. RL Training Algorithm: Actor–Critic

To generate labels, we utilize the following technique. For each sentence gold_k in the gold summary, we find its most similar sentence from the input text. We define similarity using the ROUGE-L measure. That is, $\text{label}_k = \text{argmax ROUGE-L}(\text{gold}_k, s_k)$, for each s_k in the example. We define the rewards corresponding to the agent's actions. Formally,

$$r_t = \begin{cases} \text{ROUGE-L}(s_k, \text{gold}_k), & \text{if extract} \\ \text{ROUGE-1}(\text{summ}_{\text{sys}}, \text{summ}_{\text{gold}}), & \text{if stop} \end{cases} \tag{4}$$

where summ_{sys} is the system summary and $\text{summ}_{\text{gold}}$ is the gold summary. We utilize actor–critic [30] to train our policy network.

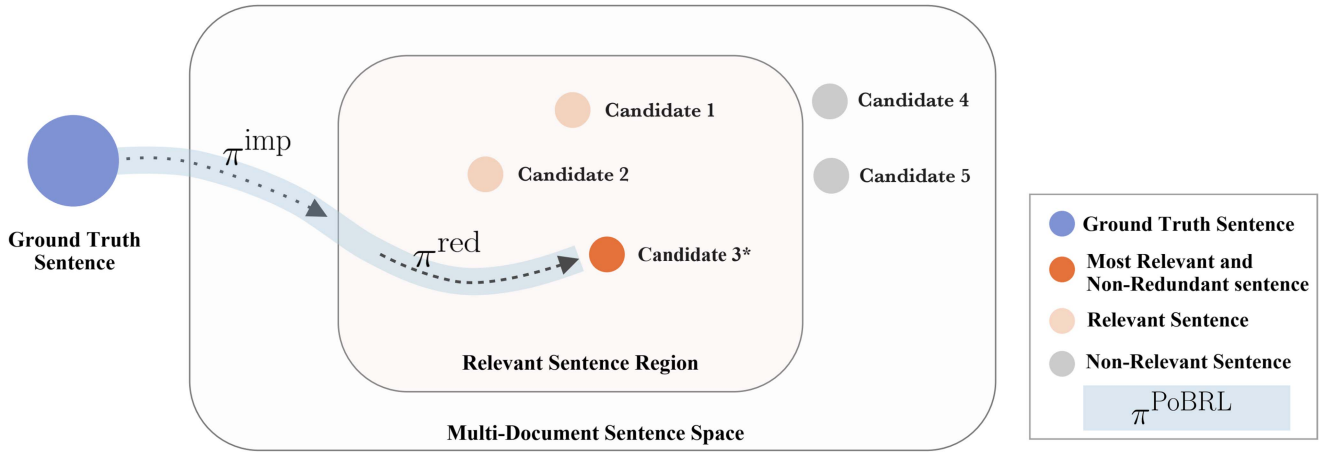


Fig. 1. Multidocument search space shows sentences across multiple related texts can represent the same ground truth sentence. The search procedure for π^{PoBRL} is to have π^{imp} identify the relevant sentence region. Then, π^{red} finds the optimal candidate sentence.

The gradient update for the weights of policy network is

$$\nabla J(\psi) = E_{\pi_\psi}[\nabla_\psi \log \pi_\psi(a_t|X_t) A(X_t, a_t)] \tag{5}$$

where $\pi_\psi(\cdot)$ is the policy function that takes an input state and outputs an action (in our case, which sentence to extract), ψ consists of the learnable parameters of the neural network. With γ as the discount factor, next, we have

$$A(X_t, a_t = s_t) = r_t + \gamma V(X_{t+1}) - V(X_t) \tag{6}$$

$$= Q(X_t, a_t = s_t) - V(X_t) \tag{7}$$

is the advantage function which measures the *advantage* of extracting a *particular sentence* s_t over the rest of the sentences in the pool, where the value function $V(X_t)$ measures the quality of the state

$$V(X_t) = E_{\pi_\theta}\left\{\sum \gamma^t r_{t+1} | X_t\right\} \tag{8}$$

and that

$$Q(X_t, a_t = s_t) = E_{\pi_\theta}\left\{\sum \gamma^t r_{t+1} | X_t, a_t = s_t\right\}. \tag{9}$$

The value function is trained by minimizing the mean squared loss of

$$\mathbb{L}_v = \sum_t ||V(X_t) - R_t||^2 \tag{10}$$

where R_t is the sum of the reward up to t.

We instantiate two sentence extraction policy networks as in Algorithm 2, each with its own objective, and trained with actor–critic.

D. Train-RL-Policy (Objective = Importance)

We train our first policy network π_θ^{imp} to maximize the *importance* over each step with actor–critic as given by (5). To do this, we modify the first part of the reward (4) to be ROUGE-L_{F_1} to encourage this RL policy network searching for the most relevant sentence region from the multidocument space.

E. Train-RL-Policy (Objective = Redundancy)

We train our second policy network π_ϕ^{red} with the *redundancy* as its objective using actor–critic. To do this, we modify the reward measure of (4) (first part) to be ROUGE-$L_{\text{Precision}}$ to encourage this policy network to identify similarity between sentences in the pool versus the ones already extracted. This in effect helps π_θ^{imp} identify the best (or the most relevant but non-redundant) sentence from the region identified by π_θ^{imp}.

IV. PoBRL Algorithmic Method

We present our PoBRL algorithmic method in Algorithm 2. We start the algorithm by independently training two policies π_θ^{imp} and π_ϕ^{red}, where θ and ϕ represent two sets of neural network parameters. We train both policies with actor–critic (Section III-C) but with different objectives. In particular, π_θ^{imp} is trained by optimizing over *importance* while the π_ϕ^{red} is trained with the *redundancy* objective.

Assume there are n articles and m sentences per article for a total of nm sentences in the input. Once we have trained these two policies, we then calculate the sentence extraction probability for each sentence in the input. We define $P_{1:nm}^{\pi} \in R^{nm \times 1}$ as the sentence extraction probability tensor with each entry representing the sentence s_k extraction probability according to policy π

$$P_{1:nm}^{\pi} \leftarrow P_\pi(s_k|X_t) \text{ for all } k \in nm. \tag{11}$$

We first calculate this sentence extraction probability from (11) with π_θ^{imp} as $P_{nm}^{\text{imp}} \leftarrow P_{\pi_\theta^{\text{imp}}}(\cdot|X_t)$, and repeat this calculation with π_ϕ^{red} as: $P_{nm}^{\text{red}} \leftarrow P_{\pi_\phi^{\text{red}}}(\cdot|X_t)$.

Now, we are ready to blend these two policies together utilizing the MMR framework, which is

$$\pi^{\text{PoBRL}}(\cdot|X_t) = \lambda \pi_\theta^{\text{imp}}(\cdot|X_t) - (1-\lambda)\pi_\phi^{\text{red}}(a_{2,t}|X_t, P_{1:nm}^{\text{imp}}) \tag{12}$$

Algorithm 2: PoBRL Algorithmic Method.

Input :document R
Output :system summary S
$S \leftarrow \{\}$
$\pi_\theta^{\text{imp}} \leftarrow$ get trained-RL-Policy(objective = importance) as in Section III-D
$\pi_\phi^{\text{red}} \leftarrow$ get trained-RL-Policy(objective = redundancy) as in Section III-E
while True **do**
  $X_t = \{R, S\}$ ▷ Update the state based on the sentences extracted.
  $P_{1:nm}^{\text{imp}} \leftarrow P_{\pi_\theta^{\text{imp}}}(\cdot|X_t)$ with π_θ^{imp} [as in (11)] ▷ Calculate the sentence extraction prob. for each sentence in R
  $P_{1:nm}^{\text{red}} \leftarrow P_{\pi_\phi^{\text{red}}}(\cdot|X_t)$ with π_ϕ^{red} [as in (11)] ▷Calculate the sentence extraction prob. for each sentence in R
  (optional) calculate a new value of λ (14) ▷Optional: we can either use a fixed λ or an adaptive λ
  formulate π^{PoBRL} by blending π_θ^{imp}, π_ϕ^{red} (12) ▷Formulate the new policy π^{PoBRL} by blending these two policies
  $P_{1:nm}^{\text{PoBRL}} \leftarrow P_{\pi^{\text{PoBRL}}}(\cdot|X_t)$ with π^{PoBRL} [as in (13)] ▷Calculate the sentence extraction prob. for each sentence in R
  $s_{\text{select}} \sim P_{1:nm}^{\text{PoBRL}}$ ▷extract one sentence by sampling from extraction probability
  $S \leftarrow S \cup \{s_{\text{select}}\}$ ▷add the extracted sentence into the set S

  if STOP $\sim \pi^{\text{PoBRL}}(.|X_t)$ **then** ▷ If the STOP action is sampled, then terminate the extraction process
    break
  end if
end while
return system summary S ▷Return the extracted summary S

where the sentence extraction probabilities of this newly blended policy PoBRL (π^{PoBRL}) are given by

$$P_{1:nm}^{\text{PoBRL}} \leftarrow P_{\pi^{\text{PoBRL}}}(\cdot|X_t) = \lambda P_{1:nm}^{\text{imp}} - (1-\lambda) P_{1:nm}^{\text{red}} \quad (13)$$

where λ is a parameter trading off the two objectives, which will be considered further in the following. Now comparing both (12) and (13) to the original MMR setup as in (1), we see that they are nearly identical. The blending of the two policies π_θ^{imp} and π_ϕ^{red} mimics (1).

Finally, to determine which sentence to extract at $t = 1$ (the first time step), we sample $s_{\text{select}} \sim \pi^{\text{PoBRL}}(\cdot|X_{t=1})$, with the sentence extraction probabilities given by (13). We append the extracted sentence s_{select} into the summary set S, and update the state $X_{t=1} = \{R, S\}$. We repeat the extraction process until a stop action occurs.

A. Adaptive Tuning of λ With the Advantage Function

One of the critical parameters of the MMR framework is λ, which trades off the importance objective versus the redundancy objective. Determining λ is a nontrivial task and there are many ways to do it. The simplest method is to set λ to be a fixed constant value. This means we will use the same λ for all input examples. Off course, this design choice is suboptimal because: 1) the best λ value depends on the content of the actual input example, and 2) the MMR is an iterative process, depending on the sentences extracted. The λ shall be *adaptive* even when extracting within one input example.

To address this problem, we propose an innovative idea that requires the following:

1) no separate offline training process;
2) can be computed online in an adaptive fashion by leveraging the power of the RL framework; and
3) incurs very little computational overhead.

When training the network with actor–critic, we get both the policy network, and the advantage function network A automatically as a byproduct. Therefore, the advantage function (7) provides a quality measure by comparing sentences from the pool.

Our solution is to utilize the advantage function, which *scores* each sentence with respect to its objective function. Specifically, when we train the network to optimize over the importance objective, then its advantage function scores the sentences with respect to the importance measure. Similarly when we train the network to optimize over redundancy, its advantage function scores the sentences with respect to the redundancy measure. Utilizing the advantage function, we have a score function that rates each sentence in accordance with the importance measure, or the redundancy/diversity measure. Importantly, because the advantage function takes the state X_t as the input (which depends on the sentences already extracted and the rest of the sentences in the document), so it is adaptive during the extraction process. Formally, we can formulate the adaptive λ as follows:

$$\lambda_t^{\text{adv}} = A^{\pi_\theta^{\text{imp}}}(X_t, a_t = s_t) - A^{\pi_\phi^{\text{red}}}(X_t, a_t = s_t) \quad (14)$$

where $A^{\pi_\theta^{\text{imp}}}$ and $A^{\pi_\phi^{\text{red}}}$ are the advantage functions of the importance and redundancy networks, respectively, and a_t defines the action of extracting a particular sentence at time t. The advantage function defined in (7) scores sentences from the pool according to the importance or redundancy measure. The motivation for this setup is that we want to measure the relative significance between the importance and the redundancy objectives [just as in the MMR framework (1)]. If the importance score (determined by $A^{\pi_\theta^{\text{imp}}}$) is higher than the redundancy score ($A^{\pi_\phi^{\text{red}}}$) (in a relative sense), then it means that the importance objective is

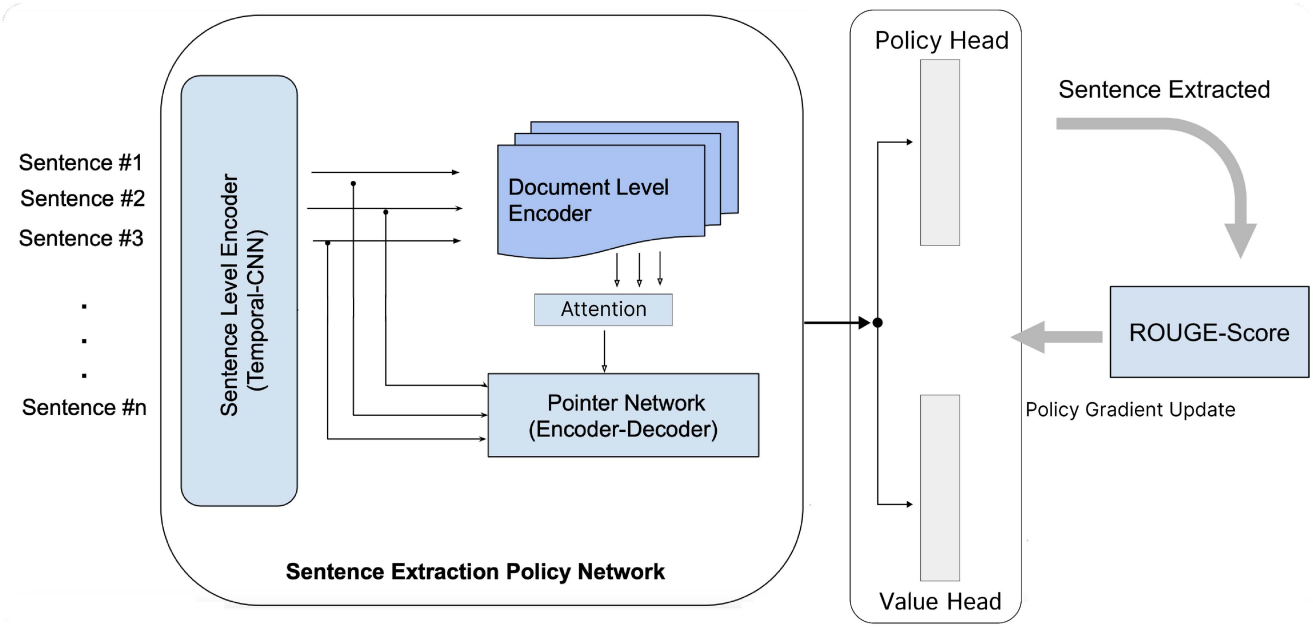


Fig. 2. Hierarchical RL Sentence Extractor Model. Each multidocument input will go through the article level path (dark blue) and the sentence level path (light blue). Then, it goes to both the policy head and the value function head. The policy head is responsible for performing the action (extracting the sentence) and it learns a distribution over each sentence (17). On the other hand, the value head learns the value function and is responsible for scoring the quality of the extracted sentence (6). Then, the rouge-score calculates the reward. Finally, the reward is used to compute the gradient via (5) and update the entire network through backpropagation.

more crucial at the extraction step (and, thus, we will have a higher λ value). Note that since the sentences already extracted are encoded by the state X_t, the advantage function is context aware.

V. Hierarchical Sentence Extractor

In this section, we describe our model structure. Our hierarchical sentence extractor consists of the following building blocks: sentence encoder, document-level encoder, attention, and pointer network. The complete model structure is shown in Fig. 2. At the high level, the model operates like a pointer network with the attention module adjusting the focus between the article level and sentence level.

The reason for designing two paths (the article-level path and the sentence aggregation path) is to incorporate a hierarchical structure into the multidocument extraction process. Since the input has the multidocument structure, we want to compare each sentence in the input against other sentences within its own document as well as the sentences in the other documents. Doing this is important since sentences across different documents might convey repeated information. To avoid excess redundancy, we not only need to extract the most critical sentences that summarize the article, but also avoid the sentences that cover similar information. Our hierarchical structure helps facilitate this process.

A. Sentence Encoder

Our model deals with each input document by reading one sentence at a time. Each input sentence will first be encoded by the sentence encoder. We use a temporal-CNN to encode each sentence, and denoted each sentence encoded hidden state with $s_{\text{enc}_i}^{\text{art}=j}$, where art $= j$ indicates the sentence that came from the jth article and i represents the sentence number from that article. Each of the encoded sentences will pass through two paths: the article-level path and the sentence aggregation path.

B. Article-Level Path

In our multidocument setting, each input contains one or more articles. Here, we are looking to formulate a representation for each article of the multidocument input. The article level path represents the dark blue colored region in Fig. 2. We implement the article-level encoder with a bidirectional long short-term memory network (LSTM). For each article of the multidocument input, this encoder takes its input sentences (e.g., $s_{\text{enc}_1}^{\text{art}=1}, s_{\text{enc}_2}^{\text{art}=1}, \ldots s_{\text{enc}_n}^{\text{art}=1}$). This article is represented by the corresponding article level sentence encoded hidden states

$$h_i^{\text{art}=j} = \text{LSTM}(s_{\text{enc}_i}^{\text{art}=j}, h_{i-1}^{\text{art}=j}), \text{ for } i \in [1, n].$$

We repeat this process for all the article of the multidocument input text.

C. Sentence Aggregate Path

In this path, we feed in each sentence $s_{\text{enc}_i}^{\text{art}=j}$ for $i \in [1, n], j \in [1, m]$ one by one by concatenating them as if they came from a single piece of text. We utilize a pointer network [31] that captures local contextual and semantic information. This pointer network utilizes information from the article level encoder and forms the basis for sentence extraction. We implement the

TABLE I
MULTINEWS DATASET

Method	Grammar	Non-redundancy	Referential clarity	Focus	Structure and Coherence
CopyTransformer	3.83	3.26	3.89	3.94	4.04
Hi-Map	4.06	3.46	3.74	4.01	3.89
MatchSum	4.26	3.32	4.12	4.00	3.96
PoBRL (Ours)	**4.28**	**4.68**	**4.19**	**4.23**	**4.08**

The bold values represent the best performances.

TABLE II
DUC-04 DATASET

Method	Rouge-1	Rouge-2	Rouge-L
Lead-3	39.41	11.77	18.03
MMR	38.77	12.91	18.61
CopyTransformer	43.57	14.03	20.50
Hi-Map	43.47	14.87	21.38
FastRL-Ext	43.56	16.05	39.69
MGSum-ext	44.75	15.75	40.84
MGSum-abs	46.00	16.81	41.36
MatchSum	46.20	16.51	41.89
RL-only	44.01	17.63	33.96
PoBRL($\lambda = 0.8$)(**Ours**)	45.13	16.69	41.12
PoBRL($\lambda = \lambda_t^{\text{adv}}$)(**Ours**)	**46.51**	**17.33**	**42.42**

The bold values represent the best performances.

pointer network with encoder–decoder-based LSTM. Formally, we write

$$u_k^t = v^T \tanh(W_{\text{sent}} e_k + W_{\text{art}} c_t) \tag{15}$$

where c_t is the output of the attention mechanism at output time t, and e_k is the hidden state of the encoder pointer network LSTM. In other words, $(e_1, e_2, e_3, ..., e_k, ..., e_{n\times m})$ are the encoder hidden states for the input sentences $s_{\text{enc}_i}^{\text{art}=j}$ for $i \in [1, n], j \in [1, m]$. Finally, v, W_{sent}, and W_{art} are the network learnable weight matrices.

D. Attention

The attention module integrates article level information to the sentence aggregate level extraction network. We implement this module with dot product attention [32].

Let d_t to be hidden state of decoder pointer network LSTM at output time index t, and h_k to be the corresponding article level sentence encoded hidden state, then

$$c_t = \sum_k \alpha_{t,k} h_k, \qquad \alpha_{t,k} = \frac{e^{(\text{score}(d_t, h_k))}}{\sum_{k'} e^{(\text{score}(d_t, h_{k'}))}} \tag{16}$$

and the $\text{score}(d_t, h_k) = d_t^T h_k$.

E. Actor–Critic Sentence Extraction Policy Network

Next, we can combine the components above to build our sentence extractor. First, we rank each candidate in the pool set by

$$P(y_k | y_{k-1}, \ldots y_0) = \text{softmax}(u_k), \text{for } k \in \text{Pool Set}. \tag{17}$$

The pool set contains all of the sentences in the input text. When a particular sentence has been extracted, we remove it from the pool set: Pool Set $\leftarrow$ Pool Set $\setminus \{s_k\}$.

F. Training Details

Training an RL neural network from scratch is difficult and time consuming because the initial policy network tends to behave randomly. To accelerate learning, we warm start it with supervised learning.

Warm start—Supervised learning: We formulate this step as a prediction problem. For a prediction target (supervised-learning training label), we follow [15]. Specifically, from the input example, we find one sentence that has the maximum ROUGE-L score compared against the ground-truth, and call that the "pseudolabel." During warm start, we train the network with supervised learning to predict the pseudolabel by minimizing the cross-entropy loss. After pretrained supervised learning, we can then train with actor–critics.

VI. EXPERIMENTATION

We showcase the performance of our methods on the following two multidocument datasets: Multi-News and DUC 2004. We begin by defining the following baselines.

1) *Lead-n:* We concatenate the first n sentences from each article to produce a system summary.
2) *MGSum (ext/abs):* This is an extractive/abstractive summarization method with multi-granularity interaction network. [10].
3) *MatchSum:* This is an extractive summarization method with text matching and BERT [7].
4) *GRU+GCN:* This is a model that uses a graph convolution network combined with a recurrent neural network to learn sentence saliency [11].
5) *MMR:* This is the MMR [2] algorithm directly applied to the multidocument to generate a system summary.
6) *OCCAMS-V:* This is a topic modeling method [12].

7) *CopyTransformer:* This is a transformer model from [33] that implements the PG network.
8) *Hi-Map:* This is a hierarchical LSTM models with MMR attention [4].
9) *FastRL-Ext:* This is the RL approach with extractive summarization focusing on single document summarization [23]. Here, we adopt it and apply it to the multidocument case.
10) *RL-only:* We train the multidocument hierarchical sentence extractor (this article) with actor–critic *without* policy blending (i.e., single policy instead of blending of two policies). This can be regarded as setting $\lambda = 1.0$ in PoBRL method [see (13) and (12)], which optimize purely on importance.

A. Experimental Setup

We report the ROUGE F_1 score with ROUGE-1, ROUGE-2, and ROUGE-SU (skip bigrams with a maximum distance of four). For our PoBRL model, we instantiate two actor–critic policy networks representing π_θ^{imp} and π_ϕ^{red}, respectively. We blend these two policies by forming π^{PoBRL} as in (12) with extraction probability in (13). For complete hyperparameter and training details, please refer to Appendix C.

Besides using a fixed λ value, we also evaluate a λ value calculated by the advantage function as in (14), and denote it by PoBRL($\lambda = \lambda_t^{\text{adv}}$). At each extraction step, the optimal λ is recalculated by balancing between importance and redundancy.

B. Empirical Performance

We empirically evaluate the performance of the models on the following two multidocument datasets.

Multi-News: Multi-News [4] is a large dataset that contains news articles scrapped over 1500 sites. The average number of words per document and ground-truth summaries are much longer than DUC-04. This dataset has 44 972 training examples and 5622 for validation and test set, respectively. We trained our models on this dataset, and report their performance in Table I. Our best model PoBRL ($\lambda = \lambda_t^{\text{adv}}$) achieves a score of 46.51/17.33/42.42, outperforming other baselines on all metrics (with the strongest ones being MatchSum and MGSum). Here, we also show performance on different values of λ. When $\lambda = 1.0$, the model reduces to a single policy and is not policy-blended (denoted by RL-only). When $\lambda = \lambda_t^{\text{adv}}$, the value of λ is adaptively determined by the advantage function as in (14). For a fixed value of $\lambda = 0.8$, we achieve a good score (45.13/16.69/41.12). However, it is intuitive that the optimal value should be varied from document to document, and it might change during the extraction process. That is reflected by the PoBRL($\lambda = \lambda_t^{\text{adv}}$) advantage method, which provides a significant performance boost. On the other hand, the RL-only represents a model without policy blend strategy. Comparing RL-only with PoBRL demonstrates the effect of our policy-blending algorithm. Incorporating policy-blending on top of this hierarchical sentence extraction model provides remarkable performance boost (going from RL-only's 44.01/15.65/33.96 to PoBRL's 46.51/17.33/42.42).

DUC-04: This is a test only dataset that contains 50 clusters with each cluster having ten documents. For each cluster, the dataset provides four human hand-crafted summaries as ground-truth. Following [4] and [8], we trained our models using the CNN-DM [34] dataset, and report the performance of different models in Table II. Our best model (PoBRL $\lambda = \lambda_t^{\text{adv}}$) achieves 38.67/10.23/13.19 (ROUGE-1/2/SU), which is a substantial improvement over all other baselines (with the closest ones being OCCAMS-V and GRU+GCN). On the other hand, using a fixed value of $\lambda = 0.8$ of PoBRL still achieves good performance (38.13/9.99/12.85). Comparing PoBRL's $\lambda = 0.8$ to $\lambda = \lambda_t^{\text{adv}}$ shows the power of our method of calculating λ with the advantage function. The effect of our policy blending algorithm is demonstrated by comparing RL-only with PoBRL. Here, we also observe a major performance boost (36.95/9.12/11.66 versus 38.67/10.23/13.19).

C. Human Evaluation

In this experiment, we randomly sampled 100 summaries of the Multi-News dataset from the following models: Hi-Map, CopyTransformer, MatchSum, and PoBRL. Following [35], we asked judges[2] (ten for each sample) to rate the quality of system generated summarization in a numerical rating of 1 to 5, with the lowest score being the worst and the highest score being the best. We evaluated the quality of the system generated summarization in terms of the following metrics as in [35] and [36]: grammar, nonredundancy, referential clarity, focus, and structure and coherence in Table III. Our PoBRL method performs the best in all five metrics.

In the multidocument setting, since an input example contains multiple articles written on a same topic, the same pieces of information are represented multiple times across one or more articles. Therefore, one challenge is repetitiveness, since the same piece of information is contained multiple times in the documents. Most importantly, in terms of nonredundancy measures, our method PoBRL significantly outperforms the rest of the group. Our method scores 4.68 out of 5.00 while the second best baseline only scores 3.46.

VII. Analysis

Since the Multi-News dataset provides a higher number and more variant/diversified input examples, this will be the dataset that we analyze below with 1000 randomly sampled examples.

A. Varying Number of Input Document

Unlike DUC04, each example of the Multi-News dataset has a different number of source documents (ranging from 2 to 9). In general, the higher the number of the example, the longer the input. Fig. 3 shows the effect of our proposed method in handling different numbers of source document in each multidocument input. As shown in the figure, the performance is almost uniformly distributed, showing that our method can handle

[2]All judges are native English speakers with at least a bachelor's degree and experience in scientific research. We compensated the judges at an hourly rate of $28.

TABLE III
HUMAN EVALUATION ON SYSTEM GENERATED SUMMARIES OF MULTI-NEWS DATASET

Method	Rouge-1	Rouge-2	Rouge-SU
Lead-1	30.77	8.27	7.35
MMR	30.14	4.55	8.16
CopyTransformer	28.54	6.38	7.22
Hi-Map	35.78	8.90	11.43
GRU+GCN:	38.23	8.48	11.56
OCCAMS_V	38.50	9.76	12.86
RL-only	36.95	9.12	11.66
PoBRL$(\lambda = 0.9)$**(Ours)**	38.13	9.99	12.85
PoBRL$(\lambda = \lambda_t^{\text{Adv}})$**(Ours)**	**38.67**	**10.23**	**13.19**

The bold values represent the best performances.

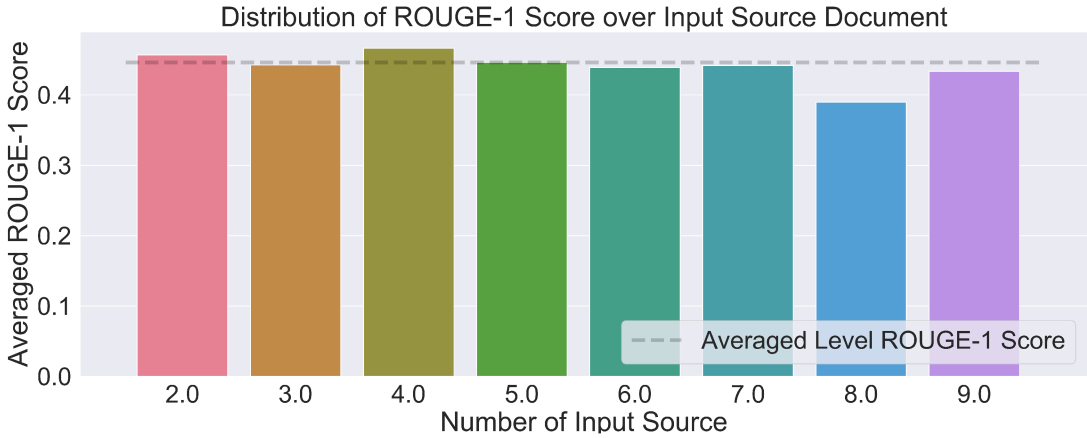


Fig. 3. Model performance on ROUGE-1 on multidocument inputs of different number of documents.

TABLE IV
REDUNDANCY ANALYSIS, THE FIRST COLUMN REDUNDANCY IS IN TERMS OF THE COSINE SIMILARITY MEASURE (THE HIGHER THE BETTER), AND THE SECOND COLUMN IS THE ROUGE-L MEASURE (THE LOWER THE BETTER)

Redundancy Measures	Cosine (Higher the better)	ROUGE-L (Lower the better)
RL-only	12.49	2.28
PoBRL	**12.73**	**0.19**

The bold values represent the best performances.

higher or lower numbers of input documents (which also translate to longer or shorter pieces of input text) equally well.

B. Redundancy

We measure the redundancy of our system summary by comparing the policy blending formulation of our model (i.e., PoBRL) against with our nonpolicy-blend (i.e., RL-only) model. This comparison helps us to understand the effectiveness of employing the policy-blending strategy. We measure redundancy by making use of the ROUGE-L score and the BERT's [37] embedding,[3] separately. Our evaluation criterion is the following: For each sentence in the generated summary, we compute the redundancy metric between that particular sentences against with the rest of the sentence in the summary. In the ROUGE-L experiment, we report the average sentence level ROUGE-L score within the input example. PoBRL reports the lowest score of 0.19 versus 2.28 as in RL-only, signifying the significant reduction in overlap (or redundancy) in the system summary. In the BERT's experiment, we report the average cosine similarity score. As given in Table IV, the PoBRL model achieves higher cosine measures (signifies more diversify and less repetitiveness).

VIII. CONCLUSION

In this article, we have proposed a novel $PoBRL$ algorithmic method that allows independent submodular reinforced policy learning and policy blending leverage on maximal marginal relevance and RL. Our empirical evaluation has shown excellent performance on several multidocument summarization datasets.

APPENDIX A
OPTIMIZATION OBJECTIVE DECOUPLING

In this section, we derive the result for decoupling the multiobjective optimization problem into small subproblems. We borrow the notation and formulation of the ROUGE score from [29].

[3]The BERT model is provided by [Online]. Available: https://github.com/google-research/bert

Let $S = \{s_i | i \leq m\}$ be a set of m sentences that constitute a system summary, and let $\rho(S)$ be the ROUGE-N score of S, where ROUGE-N evaluates the n-gram overlaps between the gold summary and the system summary S. Then,

$$\rho(S) = \frac{1}{R_N} \sum_{g \in S^*} (\min(F_s(g), F_s^*(g)) \tag{18}$$

where S^* is the gold summary, $F_s(g)$ denotes the number of times that the n-gram of g type occurs over S, and R_N denotes the number of n-gram tokens.

Let $C_{Y,S^*} = \min(F_Y(g), F_{S^*}(g))$ be the contribution of the n-gram g and $\epsilon(a \wedge b) = \sum_{g \in S^*} \max(C_{a,S^*}(g) + C_{b,S^*}(g) - F_{s^*}(g), 0)$ to be the redundancy between a and b in the summary. Then, following [29], we can write

$$\rho(S) = \sum_{i=1}^{m} \rho(s_i) \tag{19}$$

$$+ \sum_{k=2}^{m} (-1)^{k+1} \left(\sum_{1 \leq i_1 \leq \ldots \leq i_k \leq m} \epsilon^{(k)}(s_i \wedge .. \wedge s_{i_k}) \right) \tag{20}$$

$$\approx \sum_{i}^{m} \rho(s_i) - \sum_{a,b \in S, a \neq b} \tilde{\epsilon}(a \wedge b) \tag{21}$$

$$\approx \sum_{i}^{m} \text{imp}(s_i) - \sum_{a,b \in S, a \neq b} \text{red}(a, b) \tag{22}$$

where $\tilde{\epsilon}(\cdot)$ approximates the redundancy, $\text{imp}(\cdot)$ denotes the importance function, and $\text{red}(\cdot)$ denotes the redundancy function. In other words, to maximize the ROUGE score, we can optimize it by maximizing the importance of each sentence while subtracting (minimizing) over the redundancy.

Now, we define a policy π with the goal of *searching* over the sentence space and choosing the optimal set of sentences $\hat{S}^*$ such that $\pi^* = \text{argmax}_\pi E(\sum_t r_t | s_t, a_t)$, where a_t corresponds to searching over the sentence space to determine which sentence to add onto the summary S, $r_t = \rho(s_i)$ is interpreted as the importance (or the ROUGE-score contribution) of adding sentence s_i onto the summary set S, and the policy $\pi(a_t | X_t)$ outputs the probability of selecting a particular sentence and adding it onto S. Now, instead of having a predefined length m, our policy π also decides the optimal summary length by determining when to stop the selection extraction.

In the multidocument setting, however, related documents might contain large numbers of overlapping sentences. As shown in Fig. 1, for each sentence in the ground-truth summary, there exist multiple candidate sentences across related text. The optimal sentence selection strategy is to have two policies, where the first search policy π^{imp} learns to discover for the most relevant sentence space. From this smaller space, we have the second policy π^{red} that learns to search for the most relevant but nonredundant sentence (denoted as the most optimal sentence).

At a high level, these two policies π^{imp} and π^{red} are learnt using the two decomposed objectives independently as in (9). When we combine these two learned policies together [as in Algorithm 20 and (12)], these two policies complement each other and formulate the optimal search strategy π^{PoBRL}.

APPENDIX B
DETAILS ON THE TRAINING AND TESTING DATASET

We evaluate the performance of our proposed strategy on the following two datasets.

In the first experiment, we train and test on the Multi-News dataset [4]. This dataset contains 44 972 training examples, and 5622 examples for testing and also 5622 examples for validation. Each summary is written by a professional editor. We follow the testing procedure as in [4], and the dataset can be found in github link.[4]

In the second experiment, we test on the DUC-04 [36] dataset. This dataset contains 50 clusters with each cluster having ten documents. For each cluster, the dataset provide four human-generated ground-truth summaries. Because it is an evaluation-only dataset, we follow the same procedure as in [4] and [8] to train our network with the CNN-DM [34], and then test on the DUC-04 dataset. The CNN-DM dataset contains 287 113 training, 13 368 validation, and 11 490 testing examples. It can be downloaded from here,[5] and the DUC 04 dataset can be downloaded from here.[6]

APPENDIX C
TRAINING DETAILS

Our model: For our hierarchical sentence extractor, we instantiate the temporal CNN module with a setup of 1-D single layer convolutional filters of the sizes of 3, 4, and 5, respectively. Each input sentence is first converted to a vector representation by a word2vec matrix with an output dimension of 128. Then, each sentence is encoded by concatenating the output of the each window from the temporal-CNN model.

Next, we instantiate the article-level encoder with a two-layer bidirectional LSTM with 256 hidden units. We use this same configuration also for the encoder and decoder pointer network as well. For the supervised learning (as the warm start), we trained the network with an Adam optimizer with a learning rate of 1e-3 and with a batch size of 64. For the RL part, we trained the network with actor–critic and with a learning rate of 1e-7, $\gamma = 0.99$ and a batch size of 32.

The entire training time takes about 24.05 h on a T4 GPU. All hyperparameters are tuned with the validation set. For the batch size, we search over the values of [32, 128], and for the learning rate (supervised learning part), we search over the values of [1e-3, 1e-4]. For the learning rate in the RL part, we search over the values of [1e-3, 1e-7]. For hyperparameter searching, we ran one trial for one set of parameters. After selecting the values for learning rates and batch size, finally, we tune λ. Here, we only tried λ for the values of [0.8, 0.9], and we select the λ value based on its performance on the validation set. For our $\lambda = \lambda_t^{\text{adv}}$, no tuning is required.

Besides the learning rates and batch size, our overall model has one parameter (if using a fix value of λ) to tune. On the other hand, if using an adaptive $\lambda = \lambda_t^{\text{adv}}$, no tuning is required.

[4][Online]. Available: https://github.com/Alex-Fabbri/Multi-News
[5][Online]. Available: https://github.com/abisee/cnn-dailymail
[6][Online]. Available: https://duc.nist.gov/duc2004/

The overall runtime (for summarizing the Multi-News dataset) is about 8.25 min (for $\lambda = 0.8, 0.9$), and 10.5 min for $\lambda = \lambda_t^{\text{adv}}$. For the DUC-04 dataset, the runtime is about 1.80 min (for $\lambda = 0.8, 0.9$), and 2.12 min for $\lambda = \lambda_t^{\text{adv}}$.

For the *baselines*, we implemented the exact setup in the original papers.

REFERENCES

[1] J.-M. Torres-Moreno, *Automatic Text Summarization*. Washington, DC, USA: Wiley, 2014.

[2] J. Carbonell and J. Goldstein, "The use of MMR, diversity-based reranking for reordering documents and producing summaries," in *Proc. 21st Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 1998, pp. 335–336.

[3] R. McDonald, "A study of global inference algorithms in multi-document summarization," in *Proc. 29th Eur. Conf. IR Res.*, 2007, pp. 557–564.

[4] A. Fabbri, I. Li, T. She, S. Li, and D. Radev, "Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 1074–1084.

[5] H. Lin and J. Bilmes, "Multi-document summarization via budgeted maximization of submodular functions," in *Proc. Hum. Lang. Technol. Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2010, pp. 912–920.

[6] D. Gillick and B. Favre, "A scalable global model for summarization," in *Proc. Workshop Integer Linear Program. Natural Lang. Process.*, 2009, pp. 10–18.

[7] M. Zhong, P. Liu, Y. Chen, D. Wang, X. Qiu, and X. Huang, "Extractive summarization as text matching," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 6197–6208.

[8] L. Lebanoff, K. Song, and F. Liu, "Adapting the neural encoder-decoder framework from single to multi-document summarization," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 4131–4141.

[9] J. Zhang, J. Tan, and X. Wan, "Adapting neural single-document summarization model for abstractive multi-document summarization: A pilot study," in *Proc. 11th Int. Conf. Natural Lang. Gener.*, 2018, pp. 381–390.

[10] H. Jin, T. Wang, and X. Wan, "Multi-granularity interaction network for extractive and abstractive multi-document summarization," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 6244–6254.

[11] M. Yasunaga, R. Zhang, K. Meelu, A. Pareek, K. Srinivasan, and D. Radev, "Graph-based neural multi-document summarization," in *Proc. 21st Conf. Comput. Natural Lang. Learn.*, 2017, pp. 452–462.

[12] J. Conroy, S. T. Davis, J. Kubina, Y.-K. Liu, D. P. O'leary, and J. D. Schlesinger, "Multilingual summarization: Dimensionality reduction and a step towards optimal term coverage," in *Proc. Workshop Multilingual Multi-document Summarization*, 2013, pp. 55–63.

[13] Z. Cao, W. Li, S. Li, and F. Wei, "Improving multi-document summarization via text classification," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 3053–3059.

[14] M. Isonuma, T. Fujino, J. Mori, Y. Matsuo, and I. Sakata, "Extractive summarization using multi-task learning with document classification," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 2101–2110.

[15] R. Nallapati, F. Zhai, and B. Zhou, "Summarunner: A recurrent neural network based sequence model for extractive summarization of documents," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 3075–3081.

[16] C. Zheng, Y. Cai, G. Zhang, and Q. Li, "Controllable abstractive sentence summarization with guiding entities," in *Proc. 28th Int. Conf. Comput. Linguistics*, 2020, pp. 5668–5678.

[17] Y. Ma and Q. Li, "A weakly-supervised extractive framework for sentiment-preserving document summarization," *World Wide Web*, vol. 22, no. 4, pp. 1401–1425, 2019.

[18] Y. Chen, Y. Ma, X. Mao, and Q. Li, "Abstractive summarization with the aid of extractive summarization," in *Proc. Asia-Pacific Web (APWeb) Web-Age Inf. Manage. (WAIM) Joint Int. Conf. Web Big Data*, 2018, pp. 3–15.

[19] X. Li, H. Xie, Y. Song, S. Zhu, Q. Li, and F. L. Wang, "Does summarization help stock prediction? A news impact analysis," *IEEE Intell. Syst.*, vol. 30, no. 3, pp. 26–34, May/Jun. 2015.

[20] S. Narayan, S. B. Cohen, and M. Lapata, "Ranking sentences for extractive summarization with reinforcement learning," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2018, pp. 1747–1759.

[21] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3, pp. 229–256, 1992.

[22] K. Arumae and F. Liu, "Reinforced extractive summarization with question-focused rewards," in *Proc. ACL, Student Res. Workshop*, 2018, pp. 105–111.

[23] Y.-C. Chen and M. Bansal, "Fast abstractive summarization with reinforce-selected sentence rewriting," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2018, pp. 675–686.

[24] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," in *Proc. Int. Conf. Learn. Representations*, 2018.

[25] P. Li, L. Bing, and W. Lam, "Actor-critic based training framework for abstractive summarization," 2018, *arXiv:1803.11070*.

[26] Y. Dong, Y. Shen, E. Crawford, H. van Hoof, and J. Cheung, "BanditSum: Extractive summarization as a contextual bandit," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 3739–3748.

[27] R. Pasunuru and M. Bansal, "Multi-reward reinforced summarization with saliency and entailment," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2018, vol. 2, pp. 646–653.

[28] W. Czarnecki et al., "Mix and match agent curricula for reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1087–1095.

[29] M. Peyrard and J. Eckle-Kohler, "Optimizing an approximation of ROUGE - A problem-reduction approach to extractive multi-document summarization," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2016, pp. 1825–1836.

[30] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," *Adv. Neural Inf. Process. Syst.*, vol. 12, 1999.

[31] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *Proc. Adv. Neural Inf. Process. Syst.*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds., 2015, pp. 2692–2700. [Online]. Available: http://papers.nips.cc/paper/5866-pointer-networks.pdf

[32] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proc Conf. Empirical Methods Natural Langu. Process.*, 2015, pp. 1412–1421. [Online]. Available: https://www.aclweb.org/anthology/D15-1166

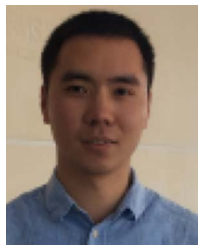
[33] S. Gehrmann, Y. Deng, and A. Rush, "Bottom-up abstractive summarization," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 4098–4109. [Online]. Available: https://www.aclweb.org/anthology/D18-1443

[34] D. Chen, J. Bolton, and C. D. Manning, "A thorough examination of the CNN/daily mail reading comprehension task," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 2358–2367. [Online]. Available: https://www.aclweb.org/anthology/P16-1223

[35] E. Chu and P. Liu, "MeanSum: A neural model for unsupervised multi-document abstractive summarization," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1223–1232.

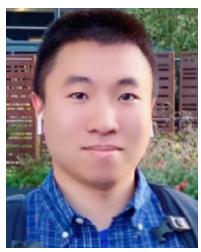
[36] H. T. Dang, "Duc 2005: Evaluation of question-focused summarization systems," in *Proc. Workshop Task-Focused Summarization Question Answering*, 2006, pp. 48–55.

[37] J. Devlin, D. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Langu. Technol.*, 2019, pp. 4171–4186.



DiJia Su is currently working toward the Ph.D. degree in machine learning with the Department of Electrical and Computer Engineering, Princeton University, Princeton, NJ, USA.

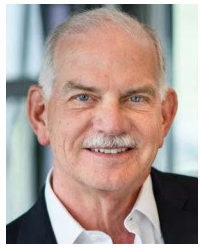
His research interests include theoretical and empirical aspects of machine learning, quantitative finance, reinforcement learning, and natural language processing.



Difei Su is currently working toward a degree with the Department of Computer Science, University of British Columbia, Vancouver, BC, Canada.

His research focuses on empirical applications of deep learning in various area, such as natural language processing.

John M. Mulvey is currently a Professor with Operations Research and Financial Engineering Department, Princeton University, Princeton, NJ, USA. He is a Founding Member of Princeton's Bendheim Center for Finance and the Center for Statistics and Machine Learning. His specialty is financial optimization and dynamic investment strategies. For more than 45 years, he has designed and implemented asset-liability management systems for numerous organizations, including PIMCO, Towers Perrin/Tillinghast, AXA, American Express, Siemens, Munich Re-Insurance, Renaissance Re-Insurance, Ant Group/Alibaba, First Republic Bank, and numerous multistrategy hedge funds, among others. His current projects address regime identification and factor approaches for long-term investors, including family offices and pension plans, with an emphasis on optimizing performance by means of goal-based investing. He has authored or coauthored more than 170 articles and edited five books, including the first implementation of a fully integrated advisor system for individual investors in 1998.



H.Vincent Poor (Life Fellow, IEEE) received the Ph.D. degree in EECS from Princeton University, Princeton, NJ, USA, in 1977.

From 1977 to 1990, he was with the faculty of the University of Illinois at Urbana–Champaign, Champaign, IL, USA. Since 1990, he has been with the faculty of Princeton University, where he is the Michael Henry Strater University Professor. From 2006 to 2016, he was the Dean of Princeton's School of Engineering and Applied Science. He has authored or coauthored a book *Machine Learning and Wireless Communications* (Cambridge University Press, 2022). His research interests include information theory, machine learning and network science, and their applications in wireless networks, energy systems, and related fields.

Dr. Poor is a Member of the National Academy of Engineering and the National Academy of Sciences, and a Foreign Member of the Chinese Academy of Sciences, the Royal Society, and other national and international academies. He was the recipient of the IEEE Alexander Graham Bell Medal in 2017.