PoBRL: Optimizing Multi-Document Summarization by Blending Reinforcement Learning Policies

Andy Su
Princeton University

Difei Su
University of British Columbia

John M. Mulvey
Princeton University

H. Vincent Poor
Princeton University

Abstract

We propose a novel reinforcement learning based framework **PoBRL** for solving multi-document summarization. PoBRL jointly optimizes over the following three objectives necessary for a high-quality summary: importance, relevance, and length. Our strategy decouples this multi-objective optimization into different sub-problems that can be solved individually by reinforcement learning. Utilizing PoBRL, we then blend each learned policies together to produce a summary that's a concise and complete representation of the original input. Our empirical analysis shows state-of-the-art performance on several multi-document datasets. Human evaluation also shows that our method produces high-quality output.

1 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 (IR) and natural language processing (NLP) domains. Traditionally, summarization can be broadly categorized as extractive or abstractive summarization (Torres-Moreno, 2014). 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 that are not in the original text.

In this paper, we tackle extractive summarization within a cluster of related texts (i.e., multi-document summarization[1]). Unlike single-document summarization, redundancy is particularly important because sentences across related documents might convey overlapping information. Thus, sentence extraction in such setting is difficult because one will need to determine which piece of information is relevant while avoiding unnecessary repetitiveness. In this sense, finding an optimal summary can be viewed as a combinatorial optimization problem. One way to solve this is the *Maximal Marginal Relevance* (MMR) algorithm (Carbonell and Goldstein, 1998). 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, 2007) the author developed a globally optimal MMR method by reformulating it as an inference problem and solving it using integer linear programming.

In this paper, we present a novel approach to tackle multi-document summarization. We formulate this as a multi-objective optimization problem and we seek to jointly optimize over the following three key metrics necessary for a high quality output (McDonald, 2007):

- Importance: only the relevant and critical information should be extracted;
- Redundancy: the extracted information should not be self-repetitive;
- Length: the final output should be as concise as possible.

Our approach optimizes these three objectives simultaneously and is able to produce output which 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 multi-document space. We present our **PoBRL** (**Po**licy **B**lending with maximal marginal relevance and **R**einforcement **L**earning) framework to decouple this multi-objective optimization problem into smaller problems that can be solved using reinforcement learning. Then through our PoBRL framework, we present a policy blending method that integrates the learned policies together so that

[1]Note: multi-document here means multiple documents about the same topic.

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 (Fabbri et al., 2019) and DUC-04 datasets, our approach demonstrates state-of-the-art results. Our **contributions** are as follows:

- We propose a novel PoBRL algorithmic framework that jointly optimize over the three identified metrics by objective decoupling, independent policy learning. We combine each learned policies by "blending";
- We propose a novel, simple, and effective multi-document summarization model that leverage the hierarchical structure of the multi-document input;
- Our empirical performance on multi-document datasets shows state-of-the-art performance. Human evaluation also shows that our method produces fluent and high-quality output.

2 Related Work

Extractive multi-document summarization can be viewed as a discrete optimization problem. In this setting, each source sentence can be regarded as binary variables. Under a pre-defined summary length as the constraint, the optimization problem needs to decide which sentences to be included in the final summary. Several techniques were introduced to solve this problem. For instance, (Lin and Bilmes, 2010) proposed to solve with maximizing sub-modular functions under budget constraint. (McDonald, 2007)(Gillick and Favre, 2009) reformulated the problem and solved 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 (Zhong et al., 2020). In (Lebanoff et al., 2018)(Zhang et al., 2018), the authors adapted the network trained on single document corpus directly to multi-document setting. In (Jin et al., 2020), the author proposed multi-granularity network for multi-document summarization. In (Yasunaga et al., 2017), this problem was solved by graph convolutional network. Other related works include (Conroy et al., 2013) (Cao et al., 2017)(Isonuma et al., 2017)(Nallapati et al., 2016).

Alternatively, there is RL approach to solve summarization. In (Shashi Narayan, Shay B. Cohen, and Mirella Lapata, 2018), the author utilized the REINFORCE (Williams, 1992) (an RL algorithm) to train the model for extractive single-doc summarization. A similar approach was also taken in (Arumae and Liu, 2018) by including a question-focus reward. In (Chen and Bansal, 2018), the author performed abstractive summarization following the sentence extraction. In (Paulus et al., 2017), REINFORCE is being applied for abstractive summarization. A more stabilize strategy (Actor-critic (Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, Yoshua Bengio, 2016)) has been considered in (Li et al., 2018). Similar approach has also been investigated in (Dong et al., 2019) (Pasunuru and Bansal, 2018). 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 optimizes for importance, and another policy optimizes for redundancy) for solving the multi-document summarization problem has not been explored in any of the NLP literature. The closest possible related work is (Czarnecki et al., 2018), in which the authors proposed a *curriculum learning based* RL method for compressing (mixing) multiples RL policies to solve a high-dimensional (**non-NLP**) task. Different than their curriculum learning approach, here we show that we can blend(mix) different policies by simply taking the MMR combination of them as in eqn.1 (and eqn.4), a much simpler and intuitive approach for solving a *NLP* task (multi-doc summarization) without introducing an extra layer of complexity.

3 Background

3.1 Maximal Marginal Relevance

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

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

where S is the produced summary, s_i is the i^{th} 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, eqn 1 is implemented as an iterative and greedy algorithm that loops through each sentence in the documents for a pre-defined summary length. The algorithm is shown in Appendix Section A.

3.2 Motivation for Reinforcement Learning

As noted by (McDonald, 2007)(Lin and Bilmes, 2010), such iterative and greedy approach is non-optimal because the decision made at each time step is local and doesn't consider its downstream consequences. The sequential nature of algorithm naturally fits within the reinforcement learning 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 (MDP). 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 pre-determined length of summary, we let our policy π to optimize it directly. We define actions of two types:

$$a_t = \begin{cases} s_t \sim P_\pi(.|X_t), & \text{if continue to extract} \\ \text{STOP extraction}, & \text{otherwise} \end{cases}$$

With this action setup, our extractive agent can design the optimal strategy as when to stop extraction. When it instead decides to continue extracting sentences, each sentence in the input will be 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 then is to come up with 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.

If we define the reward contribution to be the ROUGE score (Lin, 2004) that we wish to optimize, then it can be shown (see Appendix B) that :

$$\text{ROUGE}(S) \approx \sum_{i=1} \text{imp}(s_i) - \sum_{a,b\in S, a\neq b} \text{red}(a, b) \quad (2)$$

Fig.1 shows the sentence search process for the multi-document space. For each ground truth sentence (labeled as blue), there might exist more than one candidate sentences coming across multiple related text. For the optimal summary S^*, we want to extract the single best sentence (colored dark orange) from a pool of similar looking sentences (region denoted as"relevant sentence space"in the figure).

For effective navigation in such a search space, we proposed our PoBRL framework(Algorithm 1). Based on eqn 2, our framework decomposes the importance and redundancy objectives independently into two sub-problems. Since each problem has its own local objectives, to maximize eqn 2, we simply design policies π^{imp} and π^{red} to optimize for each objective independently. At a high level, we can think of 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 diversify (non-redundant) sentence. This search procedure is graphically illus-

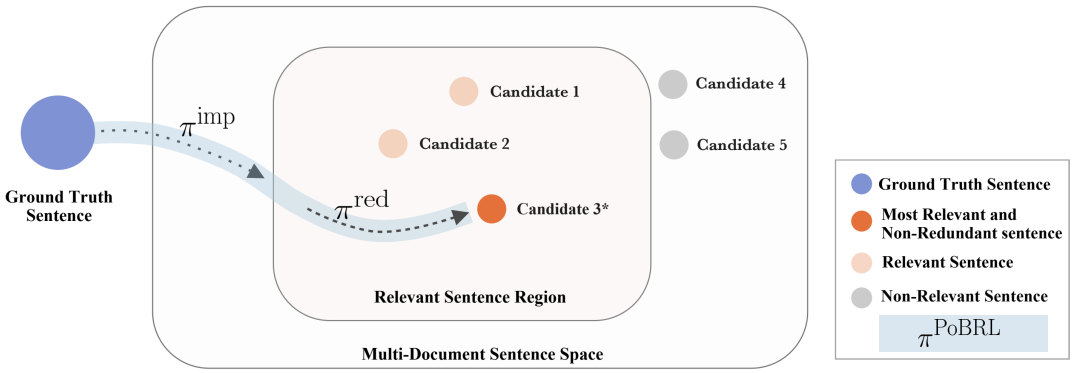

Figure 1: Multi-document search space shows sentences across multiple related texts can represent the same ground truth sentence. The search procedure for π^{PoBRL} is to have π^{imp} get to the relevant sentence region. Then from here, π^{red} finds the most optimal candidate sentence.

trated 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 most optimal sentence.

4 PoBRL Algorithmic Framework

We present our PoBRL algorithmic framework as in Algorithm 1. We start off 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 trained both policies with actor-critic (Section.4.1) but with different objective. For the π_θ^{imp}, it is trained by optimizing over *importance*. For the π_ϕ^{red}, it is trained with *redundancy* as its objective.

Algorithm 1 PoBRL Algorithmic Framework

Input: λ, R
S$\leftarrow \{\}$
$\pi_\theta^{\text{imp}} \leftarrow$train-RL-Policy(importance) as in Section 4.1.1
$\pi_\phi^{\text{red}} \leftarrow$ train-RL-Policy(redundancy) as in Section 4.1.2

while *True* **do**
  $X_t = \{R, S\}$
  $P_{1:nm}^{\text{imp}} \leftarrow P_{\pi_\theta^{\text{imp}}}(\cdot|X_t)$ with π_θ^{imp} (as in Eqn.3)
  $P_{1:nm}^{\text{red}} \leftarrow P_{\pi_\phi^{\text{red}}}(\cdot|X_t)$ with π_ϕ^{red} (as in Eqn.3)
  (optional) calculate a new value of λ (Eqn.9)
  calculate π^{PoBRL} by blending π_θ^{imp}, π_ϕ^{red} (Eqn.4)
  $P_{1:nm}^{\text{PoBRL}} \leftarrow P_{\pi^{\text{PoBRL}}}(\cdot|X_t)$ with π^{PoBRL}as in Eqn.5
  $s_{\text{select}} \sim P_{1:nm}^{\text{PoBRL}}$ (extract sentence by sampling from extraction probability) $S \leftarrow S \cup \{s_{\text{select}}\}$
  if *STOP* $\sim \pi^{PoBRL}(.|X_t)$ **then**
    break
  end
end
return system summary S

Assume there are n articles and m sentences per article for a total of nm number of sentences in the input, once we have trained these two policies, we then calculate the sentence extraction probabilities for each sentence in the input as:

$$P_{1:nm} \leftarrow P_\pi(s_k|X_t) \text{ for all } k \in nm \quad (3)$$

where $P_{1:nm} \in R^{nm\times 1}$ (with each entry represents the sentence s_k extraction probability). We first calculate this sentence extraction probability from eqn.3 with π_θ^{imp} as: $P_{1:nm}^{\text{imp}} \leftarrow P_{\pi_\theta^{\text{imp}}}(\cdot|X_t)$. We repeat this calculation with π_ϕ^{red} as: $P_{1:nm}^{\text{red}} \leftarrow P_{\pi_\phi^{\text{red}}}(\cdot|X_t)$.

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

$$\pi^{\text{PoBRL}}(\cdot|X_t) = \lambda\pi_\theta^{\text{imp}}(\cdot|X_t) - (1-\lambda)\pi_\phi^{\text{red}}(a_{2,t}|X_t, P_{1:nm}^{\text{imp}}) \quad (4)$$

where the sentences extraction probabilities of this newly blended policy PoBRL are given by:

$$P_{1:nm}^{\text{PoBRL}} \leftarrow P_{\pi^{\text{PoBRL}}}(\cdot|X_t) = \lambda P_{1:nm}^{\text{imp}} - (1-\lambda)P_{1:nm}^{\text{red}} \quad (5)$$

Now comparing both Eqn.5 and Eqn.4 to the original MMR setup as in Eqn.1, we see that they are nearly identical in terms of setup. Finally, to determine which sentence to extract at the $t = 1$ (the first time step), we sample $s_{\text{select}} \sim \pi^{\text{PoBRL}}(\cdot|X_{t=1})$, with the sentence extraction probabilities given by Eqn.5. We appended the extracted sentence s_{select} into the summary set S, and we update the state $X_{t=1} = \{R, S\}$. We repeat the extraction process until stop action is sampled.

4.1 Reinforcement Learning Training Algorithm: Actor-Critic

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

$$r_t = \begin{cases} \text{ROUGE-L}(s_k, \text{gold}_k), & \text{if extract} \\ \text{ROUGE-1}(\text{summ}_{\text{sys}}, \text{summ}_{\text{gold}}), & \text{if stop} \end{cases} \quad (6)$$

where summ$_{\text{sys}}$ is the system summary and summ$_{\text{gold}}$ is the gold summary. We utilize actor-critic(Konda and Tsitsiklis, 2003) to train our policy network. The gradient update for the weights of policy network will be:

$$\nabla J(\psi) = E_{\pi_\psi}[\nabla_\psi \log \pi_\psi(a_t|X_t) A(X_t, a_t)] \quad (7)$$

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. Next, we have

$$A(X_t, a_t = s_t) = Q(X_t, a_t = s_t) - V(X_t) \quad (8)$$

is the advantage function which measures the **advantage** of extracting a *particular sentence* s_t over the rest of the sentences in the pool. And $V(X_t) = E_{\pi_\theta}\{\sum \gamma^t r_{t+1}|X_t\}$ measures the quality of state, $Q(X_t, a_t = s_t) = E_{\pi_\theta}\{\sum \gamma^t r_{t+1}|X_t, a_t = s_t\}$.

We instantiate two sentence extraction policy networks as in Algorithm 1, each with its own objective, and trained with actor-critic.

4.1.1 train-RL-Policy(Importance)

We train our first policy network π_θ^{imp} to maximize the *importance* over each step with actor-critic as giving by eqn.7. To do this, we modify the first part of the reward eqn.6 to be ROUGE-L_{F_1} to encourage this RL policy network searching for the most relevant sentence region from the multi-document space.

4.1.2 train-RL-Policy(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 eqn.6 (first part) to be ROUGE-$L_{\text{Precision}}$ to encourage this policy network to identify similarity between each sentences in the pool versus the ones already extracted. This in effect helps π_θ^{imp} to identify the best (or the most relevant but non-redundant) sentence from the region identified by π_θ^{imp}.

4.2 Adaptive Tuning of λ with the Advantage Function

The formulation in eqn.4 and eqn.5 are good as long as we have a reasonably good λ. In practice, the λ parameter can be a fixed numerical value that can be computed offline and determined beforehand. In reality, this is not an easy task because: 1) the λ parameter might vary from document to document; and 2) the optimal value of λ might change throughout the extraction process.

To solve this problem, we propose an innovative idea that requires no separate offline training process and can be computed online in an adaptive fashion by leveraging the power of the RL framework. When we train the network with actor critic, we get the advantage function automatically as a byproduct. Therefore, the advantage function from eqn 8 provides a quality measure by comparing sentences from the pool. Utilizing the advantage function, we can have a score function that rates each sentence in accordance with the importance and redundancy/diversity balance. Formally,

$$\lambda_t^{\text{adv}} = A^{\pi_\theta^{\text{imp}}}(X_t, a_t = s_t) - A^{\pi_\phi^{\text{red}}}(X_t, a_t = s_t) \tag{9}$$

where $A^{\pi_\theta^{\text{imp}}}, A^{\pi_\phi^{\text{red}}}$ are the advantage functions of the importance and redundancy policy network respectively, and a_t defines the action of extracting a particular sentence at time t. The advantage function defined in eqn.8 scores sentences from the pool according to the imp or red measure. Note that since the sentences already extracted are encoded by the state X_t, so the advantage function is context aware.

5 Hierarchical Sentence Extractor

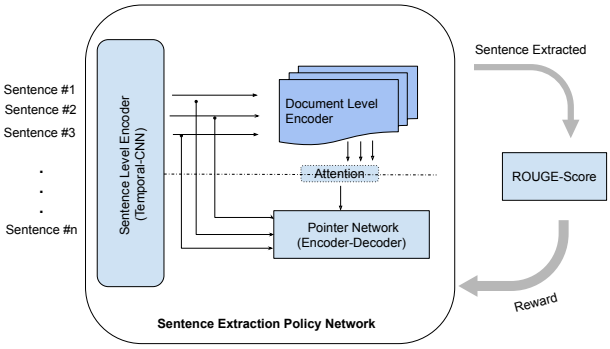

Figure 2: The Hierarchical RL Sentence Extractor Model. Each multi-document input will go through the article level path (dark blue) and the sentence level path (light blue).

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 adjusts the focus between the article level and sentence level.

5.1 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 $s_{enc_i}^{art=j}$, where $art = j$ indicates the sentence that came from the j_{th} article and i represents the sentence number from that article. Each of the encoded sentences will then go through two paths: the article-level path and the sentence aggregation path.

5.2 Article-Level Path

In our multi-document setting, each input contains one or more articles. Here, we are looking to formulate a representation for each article of the multi-document input. The article level path represents the dark blue colored region in Fig 2. We implement the article-level encoder with a bidirectional LSTM. For each article of the multi-document input, this encoder takes its input

sentences (eg, $s_{enc_1}^{art=1}$, $s_{enc_2}^{art=1}$, ...$s_{enc_n}^{art=1}$). This article is represented by the corresponding article level sentence encoded hidden states: $h_i^{art=j} = \text{LSTM}(s_{enc_i}^{art=j}, h_{i-1}^{art=j})$ for $i \in [1, n]$. We repeat this process for all the article of the multi-document input text.

5.3 Sentence Aggregate Path

In this path, we feed in each sentence $s_{enc_i}^{art=j}$ for $i \in [1, n], j \in [1, m]$ one by one by concatenating them together as if they came from a single piece of text. We utilize a pointer network (Vinyals et al., 2015) that captures local contextual and semantic information. This pointer network utilizes information by the article level encoder and form the basis for sentence extraction. We implement the pointer network with encoder-decoder based LSTM. Formally, we can write:

$$u_k^t = v^T \tanh(W_{sent} e_k + W_{art} c_t) \tag{10}$$

where c_t is the output of the attention mechanism at each output time t, 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 $s_{enc_i}^{art=j}$ for $i \in [1, n], j \in [1, m]$. Lastly, v, W_{sent}, and W_{art} are the network learn-able weight matrices.

5.4 Attention

The attention module integrates article level information to the sentence aggregate level extraction network.We implement this module with dot product attention (Luong et al., 2015).

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

$$c_t = \sum_k \alpha_{t,k} h_k, \qquad \alpha_{t,k} = \frac{e^{(\text{score}(d_t, h_k))}}{\sum_{k'} e^{(\text{score}(d_t, h_{k'}))}}, \tag{11}$$

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

5.5 Actor Critic Sentence Extraction Policy Network

Now, we can connect the dots between each component above, and this builds our sentence extractor. During the actual sentence extraction process, we ranked each candidates in the pool set by: $P(s_k|s_{k-1}, ...s_0) = \text{softmax}(u_k)$, for $k \in$ pool set. The pool set is initiated to contain all of the sentences in the input text. When a particular sentence has been extracted, we remove it from the pool set. That is: Pool Set $\leftarrow$ Pool Set $\setminus \{s_k\}$

5.6 Training Details

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

Warm Start: Supervised Learning We modify the training objective to be negative log-likelihood, and we train our network by maximizing this objective function. After pre-trained supervised learning, we can then train with actor-critics.

6 Experimentation

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

Lead-n. We concatenate the first n sentences from each article to produce a system summary.

MGSum (ext/abs), an extractive/abstractive summarization method with Multi-Granularity Interaction Network. (Jin et al., 2020).

MatchSum, an extractive summarization method with text-matching and BERT (Zhong et al., 2020).

GRU+GCN: a model that uses a graph convolution network combined with a recurrent neural network to learn sentence saliency (Yasunaga et al., 2017).

OCCAMS-V a topic modeling method (Conroy et al., 2013).

CopyTransformer This is a transformer model from (Gehrmann et al., 2018) that implements the PG network.

Hi-Map This is a hierarchical LSTM models with MMR attention (Fabbri et al., 2019).

FastRL-Ext This is the reinforcement learning approach with extractive summarization focusing on single document summarization (Chen and Bansal, 2018). Here, we adopt it and apply it to the multi-document case.

RL w/o Blend We train the multi-doc Hierarchical Sentence Extractor (this paper) 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 (eqn.5 and eqn.4) which optimize purely on importance.

6.1 Experimental Setup

We report the ROUGE F_1 score on with ROUGE-1, ROUGE-2, ROUGE-SU(skip bigrams with a maximum distance of four). For our PoBRL model, we instantiated two actor-critic policy networks representing π_θ^{imp}, π_ϕ^{red} respectively. We then blend these two polcies by forming π^{PoBRL} as in eqn.4 with extraction probability in eqn.5. For complete network hyper-parameter and training details, please refer to Appendix D.

Besides using a fixed λ value, we also evaluated a λ value calculated using the advantage function as in eqn.9, and denote it by PoBRL($\lambda = \lambda_t^{\text{adv}}$). At each extraction step, the optimal λ is recalculated by balancing between the importance and redundancy consideration.

Method	R-1	R-2	R-L
Lead-3	39.41	11.77	18.03
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 w/o Blend	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**

Table 1: MultiNews dataset

6.2 Empirical Performance

We empirically evaluate the performance of the models on the following two multi-document datasets.

Multi-News The Multi-News (Fabbri et al., 2019) is a large dataset that contains news articles scrapped over 1500 sites. The average number of words per doc and ground-truth summaries are much longer than DUC-04. This dataset has 44972 number of training examples, 5622 for validation and test set respectively. We trained our models on this dataset, and reported their performance in Table 1. Our best model PoBRL ($\lambda = \lambda_t^{\text{adv}}$) achieves a score of 46.51/17.33/42.42, outperforms other baselines in all metrics (with the strongest ones being MatchSum and MGSum). Here, we also show the performance of our model on different values of λ. When $\lambda = 1.0$, the model reduces to single policy and is not policy-blended (denoted by RL w/o Blend). When $\lambda = \lambda_t^{\text{adv}}$, the value of λ is adaptively determined by the advantage function as in eqn. 9. 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 w/o Blend represents a model without policy blend strategy. Comparing RL w/o Blend 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 w/o Blend'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 10 documents. For each cluster, the dataset provides 4 human hand-crafted summaries as ground-truth. Following (Fabbri et al., 2019)(Lebanoff et al., 2018), we trained our models using the CNN-DM (Chen et al., 2016) dataset, and report the performance of different models on Table 3. 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 w/o Blend with PoBRL. Here, we also observe a major performance boost (36.95/9.12/11.66 versus 38.67/10.23/13.19).

6.3 Human Evaluation

In this experiment, we randomly sampled 100 summaries of Multi-News dataset from the following models: Hi-Map, CopyTransformer, MatchSum, and PoBRL. Following (Chu and Liu, 2019), we asked judges[2] (10 for each sample) to rate the quality of system generated system in a numerical rating of 1 to 5, with the lowest score to be worst and the highest score to be the best. We evaluated the quality of the system generated summarization in terms of the following metrics as in (Chu and Liu, 2019) (Dang, 2006): Grammatically, Non-redundancy, Referential clarity, Focus, Structure

[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.

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**

Table 2: Human evaluation on system generated summaries of Multi-News dataset

and Coherence. Our result is shown in Table 2.

Method	R-1	R-2	R-SU
Lead-1	30.77	8.27	7.35
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 w/o Blend	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**

Table 3: DUC-04 dataset

Redundancy Measures	Cosine (Higher the better)	ROUGE-L (Lower the better)
RL w/o Blend	12.49	2.28
PoBRL	**12.73**	**0.19**

Table 4: Redundancy Analysis, the first column redundancy is in cosine similarity measure (the higher the better), and the second column is the ROUGE-L measure (the lower the better).

7 Analysis

Since the Multi-news dataset provides higher number and more variant/diversified input examples, this will be the dataset which we will be analyzing below with 1000 examples randomly sampled.

7.1 Varying number of input document

Unlike DUC04, each example of the multi-news dataset has different number of source documents (range from 2 to 9). In general, the higher the number of the example, the longer the input. In Fig 3, we show the effect of our proposed method in handling different numbers of source document in each multi-doc input. As can be seen in the figure, the performance is almost uniformly distributed, showing that our method can handle higher or lower number of input documents (which also translate to longer or shorter piece of input text) equally well.

7.2 Redundancy

We measure the redundancy of our system summary by comparing the policy blending formulation of our model (i.e., PoBRL) against with our non-policy-blend (i.e., RL w/o Blend) model. This comparison allows 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 (Devlin et al., 2019) embedding, separately. Our evaluation criterion is the following: for each sentence in the generated summary, we compute the redundancy metric between that particular sentence against with the rest of the sentence in the summary. In ROUGE-L experiment, we report average sentence level ROUGE-L score within the input example. PoBRL reports the lowest score of 0.19 versus 2.28 as in RL w/o Blend, 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 shown in Table 4, the PoBRL model achieves higher cosine measure (signifies more diversify and less repetitive).

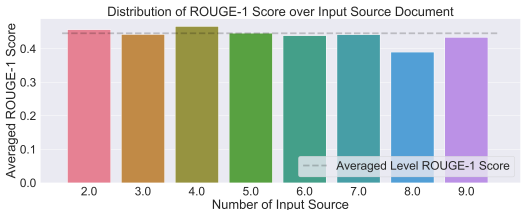

Figure 3: Model performance on ROUGE-1 on Multi-doc inputs of different number of document.

8 Conclusion

In this paper, we have proposed a novel **PoBRL** algorithmic framework that allows independent sub-modular reinforced policy learning and policy blending leverage on maximal marginal relevance and reinforcement learning. We have verified the proposed framework on our proposed model. Our empirical evaluation shows state-of-the-art performance on several multi-document summarization datasets.

References

Kristjan Arumae and Fei Liu. 2018. Reinforced extractive summarization with question-focused rewards. In *Proceedings of ACL 2018, Student Research Workshop*, pages 105–111, Melbourne, Australia. Association for Computational Linguistics.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2017. Improving multi-document summarization via text classification. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 3053–3059. AAAI Press.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA. Association for Computing Machinery.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the CNN/daily mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367, Berlin, Germany. Association for Computational Linguistics.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. *ArXiv*, abs/1805.11080.

Eric Chu and Peter Liu. 2019. MeanSum: A neural model for unsupervised multi-document abstractive summarization. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1223–1232, Long Beach, California, USA. PMLR.

John Conroy, Sashka Davis, Jeff Kubina, Yi-Kai Liu, Dianne O'leary, and Judith Schlesinger. 2013. Multilingual summarization: Dimensionality reduction and a step towards optimal term coverage.

Wojciech Czarnecki, Siddhant Jayakumar, Max Jaderberg, Leonard Hasenclever, Yee Whye Teh, Nicolas Heess, Simon Osindero, and Razvan Pascanu. 2018. Mix and match agent curricula for reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1087–1095, Stockholmsmässan, Stockholm Sweden. PMLR.

Hoa Trang Dang. 2006. Duc 2005: Evaluation of question-focused summarization systems. In *Proceedings of the Workshop on Task-Focused Summarization and Question Answering*, SumQA '06, page 48–55, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2019. Banditsum: Extractive summarization as a contextual bandit.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, Yoshua Bengio. 2016. An actor-critic algorithm for sequence prediction. arXiv:1503.06733. Version 2.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, Florence, Italy. Association for Computational Linguistics.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, Brussels, Belgium. Association for Computational Linguistics.

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 10–18, Boulder, Colorado. Association for Computational Linguistics.

Masaru Isonuma, Toru Fujino, Junichiro Mori, Yutaka Matsuo, and Ichiro Sakata. 2017. Extractive summarization using multi-task learning with document classification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2101–2110, Copenhagen, Denmark. Association for Computational Linguistics.

Hanqi Jin, Tianming Wang, and Xiaojun Wan. 2020. Multi-granularity interaction network for extractive and abstractive multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6244–6254.

Vijay R. Konda and John N. Tsitsiklis. 2003. On actor-critic algorithms. *SIAM J. Control Optim.*, 42(4):1143–1166.

Logan Lebanoff, Kaiqiang Song, and Fei Liu. 2018. Adapting the neural encoder-decoder framework from single to multi-document summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4131–4141, Brussels, Belgium. Association for Computational Linguistics.

Piji Li, Lidong Bing, and Wai Lam. 2018. Actor-critic based training framework for abstractive summarization.

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

Hui Lin and Jeff Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 912–920, Los Angeles, California. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *Proceedings of the 29th European Conference on IR Research*, ECIR'07, page 557–564, Berlin, Heidelberg. Springer-Verlag.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2016. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents.

Ramakanth Pasunuru and Mohit Bansal. 2018. Multi-reward reinforced summarization with saliency and entailment.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization.

Maxime Peyrard and Judith Eckle-Kohler. 2016. Optimizing an approximation of ROUGE - a problem-reduction approach to extractive multi-document summarization. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1825–1836, Berlin, Germany. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization. In *NAACL*, pages 33–40.

Juan-Manuel Torres-Moreno. 2014. *Automatic Text Summarization*. Wiley, Washington, DC.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2692–2700. Curran Associates, Inc.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. 2017. Graph-based neural multi-document summarization.

Jianmin Zhang, Jiwei Tan, and Xiaojun Wan. 2018. Adapting neural single-document summarization model for abstractive multi-document summarization: A pilot study. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 381–390, Tilburg University, The Netherlands. Association for Computational Linguistics.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching.

A An Iterative and Greedy MMR Algorithm

Algorithm 2 Greedy Iterative MMR Algorithm

Result: System summary
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
    if *MMR-Score* $<$ λ *score*$_{imp}$ $-$ $(1-\lambda)$*score*$_{red}$ **then**
      $s_{\text{pick}} = s_i$
      Update MMR-Score
    end
  end
  $S \leftarrow S \cup \{s_{\text{pick}}\}$
end

The MMR from equation 1 is implemented as an iterative and greedy algorithm. The complete procedure is shown in Algorithm 2.

B Optimization Objective Decoupling

In this section, we derive the result for decoupling the multi-objective optimization problem into small sub-problems. We borrow the notation and formulation of ROUGE score from (Peyrard and Eckle-Kohler, 2016).

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,

$$\rho(S) = \frac{1}{R_N} \sum_{g \in S^*} (min(F_s(g), F_s^*(g)) \quad (12)$$

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 $C_{Y,S^*} = min(F_Y(g), F_{S^*}(g))$ to be the contribution of the n-gram g, $\epsilon(a \wedge b) = \sum_{g \in S^*} max(C_{a,S^*}(g) + C_{b,S^*}(g) - F_{s^*}(g), 0)$ to be the redundancy between between a, b in the summary, it can be shown (Peyrard and Eckle-Kohler, 2016) that we can write

$$\rho(S) = \sum_{i=1}^{m} \rho(s_i) \quad (13)$$

$$+ \sum_{k=2}^{m} (-1)^{k+1} \Big(\sum_{1 \leq i_1 \leq \ldots \leq i_k \leq m} \epsilon^{(k)}(s_i \wedge .. \wedge s_{i_k}) \Big) \quad (14)$$

$$\approx \sum_{i}^{m} \rho(s_i) - \sum_{a,b \in S, a \neq b} \tilde{\epsilon}(a \wedge b) \quad (15)$$

$$\approx \sum_{i}^{m} \text{imp}(s_i) - \sum_{a,b \in S, a \neq b} \text{red}(a, b) \quad (16)$$

where $\tilde{\epsilon}(\cdot)$ approximates the redundancy, imp$(\cdot)$ denotes importance function, and red$(\cdot)$ denotes redundancy function. In other word, 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 picking the most 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 be added 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 pre-defined length m, our policy π also decides the optimal summary length by determining when to stop the selection extraction.

In the multi-document setting, however, related documents might contain large amount of overlapping sentences. As shown in Fig.1, for each sentence in the ground-truth summary, there exist multiple candidate sentence across related text. The optimal sentence selection strategy is to have two policies, with the first search policy π^{imp} learns to discover for the most relevant sentence space. And from this smaller space, we have the second policy π^{red} learns to search for the most relevant but non redundant sentence (denotes as the most optimal sentence).

At a high level, these two policies, with π^{imp} and π^{red} are learnt with the two decomposed objective independently as in equation 9. When we combine these two learned policies together (as in Algorithm 1, and eqn.4), these two policies complement each other and formulate the optimal search strategy π^{PoBRL}.

C 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 (Fabbri et al., 2019). This dataset contains 44972 training examples, and 5622 examples for testing and also 5622 examples for validation. Each summary is written by professional editor. We follow the testing procedure as in (Fabbri et al., 2019), and the dataset can be found in github link [3].

In the second experiment, we test on the DUC-04 (Dang, 2006) dataset. This dataset contains 50 clusters with each cluster having 10 documents. For each cluster, the dataset provide 4 human generated ground-truth summaries. Because it is a evaluation-only dataset, we follow the same procedure as in (Fabbri et al., 2019)(Lebanoff et al., 2018) to train our network with the CNN-DM (Chen et al., 2016), 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 [4], and the DUC 04 dataset can be downloaded from here [5].

D 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 size of 3, 4, and 5, respectively. Each input sentence is first converted to a vector representation by a word2vec matrix with a 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 2 layers bidirectional LSTM with 256 hidden units. We use this same configuration also for the encoder and decoder pointer-network as well. For the supervise learning (as the warm start), we trained the network with Adam optimizer with a learning rate of 1e-3 and with a batch size of 64. For the reinforcement learning 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 hours on a T4 GPU. All hyper-parameters 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 reinforcement learning part, we search over the values of [1e-3, 1e-7]. For hyperparmeters searching, we ran 1 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 base 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 1 parameter (if using a fix value of λ) to tune. On the other hand, if using a adaptive $\lambda = \lambda_t^{\text{adv}}$, no tuning is required.

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

For the **baselines**, we used the exact setup as in their original papers.

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