Differentiable Multi-Agent Actor-Critic for Multi-Step Radiology Report Summarization

Sanjeev Kumar Karn[1], Ning Liu[2], Hinrich Schütze[3] and Oladimeji Farri[1]

[1]Digital Technology and Innovation, Siemens Healthineers, Princeton
[2]Corporate Technology, Siemens AG, Beijing
[3]Center for Information and Language Processing (CIS), LMU Munich
{sanjeev.kumar_karn,oladimeji.farri}@siemens-healthineers.com
liuning@siemens.com inquiries@cislmu.org

Abstract

The IMPRESSIONS section of a radiology report about an imaging study is a summary of the radiologist's reasoning and conclusions, and it also aids the referring physician in confirming or excluding certain diagnoses. A cascade of tasks are required to automatically generate an abstractive summary of the typical information-rich radiology report. These tasks include acquisition of salient content from the report and generation of a concise, easily consumable IMPRESSIONS section. Prior research on radiology report summarization has focused on single-step end-to-end models – which subsume the task of salient content acquisition. To fully explore the cascade structure and explainability of radiology report summarization, we introduce two innovations. First, we design a two-step approach: extractive summarization followed by abstractive summarization. Second, we additionally break down the extractive part into two independent tasks: extraction of salient (1) sentences and (2) keywords. Experiments on English radiology reports from two clinical sites show our novel approach leads to a more precise summary compared to single-step and to two-step-with-single-extractive-process baselines with an overall improvement in F1 score of 3-4%.

1 Introduction

A diagnostic radiology report about an examination includes FINDINGS in which the radiologist describes normal and abnormal imaging results of their analysis (Dunnick and Langlotz, 2008). It also includes IMPRESSIONS or a summary that communicates conclusions about the findings and suggestions for the referring physician; a sample report is shown in Table 1. FINDINGS are often lengthy and information-rich. According to a survey of referring physicians, IMPRESSIONS may be the only part of the report that is read (Wallis and McCoubrie, 2011). Overall, referring physicians seem to appreciate the explainability (or self-explanitoriness) of IMPRESSIONS as it helps them evaluate differential diagnoses while avoiding additional conversations with the radiologist or the need for repeat procedures.

FINDINGS
$^{\psi}$there is no evidence of *midline shift* or *mass effect*.
there is soft tissue swelling or hematoma in the right frontal or supraorbital region.
underlying sinus walls and calvarium are intact.
there is no obvious *laceration*.
$^{\psi}$there is subtle thickening of the *falx* at the high convexity with its mid to posterior portion.
there is no associated subarachnoid hemorrhage.
$^{\psi}$this likely reflects normal prominence of the *falx* in a patient of this age.
$^{\psi}$remote consideration would be a very thin *subdural collection*.
IMPRESSIONS
1) no definite acute intracranial process. 2) mild prominence of the falx is likely normal for this patient. 3) remote possibility of very thin subdural collection has not been entirely excluded.

Table 1: FINDINGS (top) and IMPRESSIONS (bottom) sections of a radiologist's report. ψ indicates a sentence in FINDINGS that overlaps with sentences in IMPRESSIONS. Italicized words in FINDINGS are core concepts (e.g., disorder and procedure) that assist in answering clinical questions.

A well known end-to-end method for text summarization is *two-step*: extractive summarization followed by abstractive summarization. For instance, Chen and Bansal (2018) initially train extractive and abstractive systems separately and then use the extractive system as an agent in a single-agent reinforcement learning (RL) setup with the abstractive system as part of the environment. Their extractive system extracts salient sentences and the abstractive system paraphrases these sentences to produce a summary. This summary is in turn used to compute the reward for RL training. However, this single-agent setup often fails to extract some

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 1542 - 1553

salient sentences or it extracts irrelevant ones, leading to the generation of incomplete/incorrect IMPRESSIONS. We hypothesize that granular categories of core concepts (e.g., abnormalities, procedures) can be leveraged for generating more comprehensive summaries. Thus, a separate RL agent is dedicated to the task of extracting salient keywords (core concepts) in the two-step system. The novelty in this approach is that the new, second agent can now collaborate with the first one and the two can influence each other in their extraction decisions.

Multiagent reinforcement learning (MARL) requires that an agent coordinate with the other agents to achieve the desired goal. MARL often has centralized training and decentralized execution (Foerster et al., 2016; Kraemer and Banerjee, 2016). There are several protocols for MARL training, such as sharing parameters between agents and explicit (Foerster et al., 2016, 2018; Sukhbaatar et al., 2016; Mordatch and Abbeel, 2018) or implicit (Tian et al., 2020) communication between agents by using an actor-critic policy gradient with a centralized critic for all agents (Foerster et al., 2018). The aim of these protocols is to correctly assign credits so that an agent can deduce its contribution to the team's success. To train our cooperative agents that extract salient sentences and keywords, we propose a novel Differentiable Multiagent Actor-Critic (DiMAC) RL learning method. We learn independent agents in an actor-critic setup and use a communication channel to allow agents to coordinate by passing real-valued messages. As gradients are pushed through the communication channel, DiMAC is end-to-end trainable across agents.

The novelties in the paper are threefold:

- a summarization system that leverages core concepts via keywords, refines them and makes them the basis for more fine-grained explainability
- a multi-agent RL (MARL) based extractive component for a two-step summarization framework,
- a Differentiable Multi-agent Actor-Critic (DiMAC) with independent actors leveraging a communication channel for cooperation

The remaining paper is structured as follows. In Section 2, we provide a detailed description of our two-step framework. In Section 3, we introduce the DiMAC training algorithm. In Section 4, we describe training data and experiments. In Section 5, we discuss the results. In Section 6, we discuss related work. In Section 7, we present our conclusions.

Notations					
General		**MLE**		**RL**	
F	FINDINGS	E	encoder network	a	agent (actors)
I	IMPRESSIONS	D	pointer network	c	critic
K	Keywords	$w2w$	word LSTM	u	action
w	word	$s2s$	sentence LSTM	m	message value
s	sentence	α	attention score	r	reward
p	position	c	context vector	G	discounted reward
m	total words	v	trainable vector	V	value function
n	total sentences	q	switch value	Q	action value function
h	hidden vector	W	trainable matrix	A	advantage function
y	train label	$Conv$	CNN network		

Table 2: Notation used in this paper: general notation and notation for two-step maximum likelihood estimation (MLE) and reinforcement learning (RL). Notations are often combined, e.g., $h^{E_{w2w}}$ refers to the word encoder's hidden state vector and a_w to the word agent.

2 Model

Problem statement. We design a two-step summarization framework that takes the FINDINGS (F) section of a radiology report (consisting of a sequence of sentences) and a set of keywords (K) as input and produces an IMPRESSIONS (I) section (consisting of a sequence of sentences).

In the first step of the framework, the two extractors independently select words and sentences from FINDINGS F but also coordinate such that the selection of salient words is followed by the selection of the sentence comprising these words. In the next step, a seq2seq abstractor paraphrases the selected sentences to generate IMPRESSIONS I. Figure 1 illustrates the proposed framework. We refer to Table 2 for basic notations used in this paper. We often combine notations to indicate a framework component concisely.

Two-step summarization framework. The proposed system includes encoder networks to encode words and sentences into vector representations. It also includes two pointer extractor networks (Vinyals et al., 2015) to determine salient words and sentences by selecting their indices. Both extractor networks run for the same number of steps; however, at each step, the output index of one extractor network is chosen while the other is set as empty ($\emptyset$). When the input is $\emptyset$, an extractor pauses its activity and guides the other extractor in an optimal direction.

Encoder. A bi-directional LSTM based word

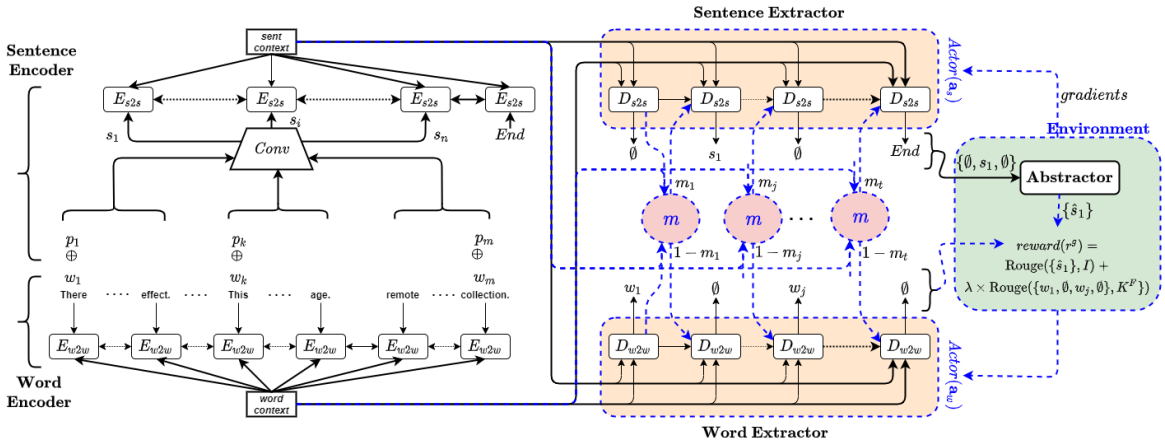

Figure 1: Our two-step summarization framework. DiMAC components (actors/extractors, communicator (m), environment and communication between them) are indicated by blue dashed lines and arrows. (i) The first step of the framework consists of encoder-extractor networks. Left side: sentence (E_{s2s}) and word (E_{w2w}) encoders. Right side: sentence (D_{s2s}) and word (D_{w2w}) extractors. Word and sentence encoders are bi-directional LSTMs with word (v^w) and sentence (h^s) embeddings as input. A convolutional network (*Conv*) obtains a sentence embedding (h^s) from word (v^w) and position (v^p) embeddings. An extractor is an LSTM pointer network with context vectors as input and either empty ($\emptyset$) or a source position as output at each step. (ii) In the second step of the framework, the seq2seq abstractor paraphrases selected sentences. During DiMAC reinforcement learning, the communicator takes contexts and actor hidden states and sends them back messages (m). The critic is omitted. Abstracted sentences ($\hat{s}$) and selected words are used to compute rewards. Figure best viewed in color.

encoder, E_{w2w}, is run on m word embeddings of FINDINGS sentences to obtain word representations, $\{h_1^{E_{w2w}}, \cdots, h_m^{E_{w2w}}\}$. A convolutional network ($Conv$) is run on concatenated word (v^w) and position (v^p) embeddings in a sentence to obtain an intermediate sentence representation (h^s). Then, a bi-directional LSTM sentence encoder, E_{s2s}, leverages the intermediate representations to obtain the final sentence representations, $\{h_1^{E_{s2s}}, \cdots, h_n^{E_{s2s}}\}$.

Extractors. Two LSTM based pointer extractors, i.e., word, D_{w2w}, and sentence, D_{s2s}, select a source word and sentence index at each step of decoding respectively. At any step j of decoding, each extractor independently uses its hidden state $h_j^{D_{w2w}}$ and $h_j^{D_{s2s}}$ to compute an attention score over its source item w_i and s_k as:

$$\alpha_{i,j}^w = \text{softmax}(v^T \phi(W^D h_j^{D_{w2w}} + W^E h_i^{E_{w2w}}))$$

$$\alpha_{k,j}^s = \text{softmax}(\hat{v}^T \phi(\hat{W}^D h_j^{D_{s2s}} + \hat{W}^E h_k^{E_{s2s}}))$$

where W^D, W^E, v, $\hat{W}^D$, $\hat{W}^E$ and $\hat{v}$ are trainable parameters, T and ϕ are transpose and $\tanh$ functions respectively, and softmax normalizes the scores. Word and sentence context vectors are computed using attention scores and encoder representations as $c_j^w = \sum_{i=1}^m \alpha_{i,j}^w h_i^{E_{w2w}}$ and $c_j^s = \sum_{k=1}^n \alpha_{k,j}^s h_k^{E_{s2s}}$ respectively.

Additionally, at step j, the decision on whether word or sentence extractor output is set to $\emptyset$ is based on a switch probability $q_j = \sigma(\text{switch}(h_j^{D_{w2w}}, c_j^w, h_j^{D_{s2s}}, c_j^s))$, where switch is a feed-forward network (omitted in Figure 1). The switch value of 0 or 1 indicates whether to set the output of sentence or word extractor to $\emptyset$.

Based on its current cell state $h_j^{D_{s2s}}$, D_{s2s} computes the next cell state, both the context vectors c_j^w and c_j^s and the selected source item encoder representation, $h_\cdot^{E_{s2s}}$. Sharing context vectors between extractors is similar to the cross attention mechanism as described by Jadhav and Rajan (2018). In case D_{s2s} is at pause (i.e., q_j=0), the E_{s2s} end representation is taken as the selected item representation. D_{w2w} follows the same approach to compute its next state.

As we lack gold standard FINDINGS keywords and sentence-wise one-to-one match between IMPRESSIONS and FINDINGS to train networks to perform selection, we heuristically obtain such labels. See Section 4.2 for details. We perform a maximum-likelihood (ML) end-to-end training of the encoder-extractor networks to minimize the following loss; $\sum_{j=1}^t -(1 - y_j^q)(y_j^w \log \alpha_j^w) - y_j^q(y_j^s \log \alpha_k^s) - y_j^q \log q_j$, where t is the step when D_{s2s} selects a dummy *END*, which indicates end of the extraction, and y_j^q, y_j^s and y_j^w are heuristi-

cally obtained switch, word and sentence selection labels at step j respectively.

Abstractor. The abstractor condenses each selected sentence to a concise summary. We employ a pointer generator network (See et al., 2017) for this purpose. It uses a copy mechanism to solve the out-of-vocabulary (OOV) problem and a coverage mechanism to solve the repetition problem. See (See et al., 2017) for details. We independently train the abstractor using heuristically obtained one-to-one matches between FINDINGS and IMPRESSIONS sentences.

3 DiMAC

As extractor and abstractor are separately trained in a two-step framework, Chen and Bansal (2018) proposed using RL training with the extractor assuming the agent role and the abstractor as part of the environment to address the separation. Furthermore, as RL loss is computed out of final summary and ground-truth IMPRESSIONS, RL training addresses the error due to heuristic labels in the pretrained networks. Unlike Chen and Bansal (2018), our setup involves multiple extractors, so we use MARL for the coordination. In other words, the word and sentence extractors D_{w2w} and D_{s2s} operate as RL agents a_w and a_s (Figure 1, right sidie).

In (Foerster et al., 2018), an actor-critic MARL has a centralized critic and parameter-sharing actors. In contrast, our extractors have different characteristics, e.g., amount of selection (salient words greater than sentences) and size of source representations; therefore, we exclude parameter sharing between actors. Additionally, to not have actors influence each other's policies, we have a critic that estimates the value function by not conditioning on the actions of other agents, thereby ensuring actor independence. Furthermore, we introduce a communicator (m) that coordinates actors through message passing. The dedicated channel m addresses the issue of the environment appearing non-stationary due to independent agents; see (Foerster et al., 2016; Sukhbaatar et al., 2016; Mordatch and Abbeel, 2018). The channel allows gradients to flow between actors, transforming the setup into an end-to-end Differentiable Multi-agent Actor Critic (DiMAC). The actors and the communicator are initialized with the maximum-likelihood (ML) trained extractors and switch network, respectively.

Actions. Actors a_w and a_s have action spaces of source words $\{w_1, \cdots, w_m\}$ and sentences $\{s_1, \cdots, s_n\}$, respectively. At any decoding step j, actors choose actions (i.e., source selection) $u_j^{a_w}$ and $u_j^{a_s}$ by using policy networks π^{a_w} and π^{a_s} and hidden states $h_j^{a_w}$ and $h_j^{a_s}$. Due to the communication between actors in DiMAC training, we intuitively expect some correlation in the actions.

Reward. For any decoding step j, if the communicator indicates sentence selection ($m = 0$), a sentence reward $r_j^{a_s}$ is computed using R_1 (ROUGE unigram recall) between the abstract summary $\hat{s}_j^{a_s}$ of selected sentence $s_j^{a_s}$ (out of action $u_j^{a_s}$) from the abstractor and a ground-truth IMPRESSIONS sentence. We sequentially match summary and IMPRESSIONS sentences such that a_s learns to select relevant sentences sequentially. Similarly, word reward $r_j^{a_w}$ for selected word $w_j^{a_w}$ out of action $u_j^{a_w}$ is 1 if the word is in the subset of keywords in FINDINGS, K^F, else it is 0. Again, we match selected and FINDINGS keywords sequentially. When an agent selects extra items, the reward for those selections is 0, and thus, the agent learns to select only relevant sentences and keywords.

In addition, joint actions of actors eventually generate a global reward in a multi-agent cooperative setting as: $r^g = R_1(\{\hat{s}_1^{a_s}, \cdots, \emptyset, \cdots, \hat{s}_t^{a_s}\}, I) + \lambda R_1(\{w_1^{a_w}, \cdots, \emptyset, \cdots, w_t^{a_w}\}, K^F)$, where t is the step when a_s selects *END* and λ is a hyperparameter to adjust the global word reward contribution. As K^F keywords are not gold-standard, we set $\lambda = 0.1$; this means that generated summary sentences drive most of the global learning. r^g is included as the reward at the last step t for both actors.

Action value functions $Q_j^{a_w}$ and $Q_j^{a_s}$ for actions $u_j^{a_w}$ and $u_j^{a_s}$ are estimated as $\mathbb{E}_{u_{j:t}^{a_w}, h_{j:t}^{a_w}}[G_j^{a_w} \mid h_j^{a_w}, u_j^{a_w}]$ and $\mathbb{E}_{u_{j:t}^{a_s}, h_{j:t}^{a_s}}[G_j^{a_s} \mid h_j^{a_s}, u_j^{a_s}]$, respectively, where $G_j^{a_w}$ and $G_j^{a_s}$ are discounted rewards computed as $\sum_{l=0}^{t-j} \gamma^l r_{j+l}^{a_w}$ and $\sum_{l=0}^{t-j} \gamma^l r_{j+l}^{a_s}$ and $\gamma = 0.99$ is a hyperparameter.

Critic. Like the actors, the critic c is an LSTM based network. It runs for the same number of steps as the actors and estimates gradients to train them. As the critic is used only in training, at each step j, the critic conditions on the actors' ground-truth selection indices, y_j^s and y_j^w, as the actions and uses these indices to obtain word and sentence encoder representations. In addition to source representations, it uses its state, h_j^c, and attends to all encoder states, $\{h_1^{E_{w2w}}, \cdots\}$ and $\{h_1^{E_{s2s}}, \cdots\}$) to estimate a value function V_j. V_j is then used to compute advantage functions $A_j^{a_w}$

Algorithm 1 Differentiable Multi-Agent Actor Critic

procedure TRAIN-DIMAC
  Initialize parameters of actors (a_w and a_s), critic (c) & communicator (m) as $\theta^{a_s} := D_{s2s}$, $\theta^{a_w} := D_{w2w}$, $\theta^c := D_{w2w}$ & $\theta^m := switch$
  for each training episode *i* **do**
    step $j \leftarrow 1$
    while action $u_j^{a_s} \neq$ END **do**
      compute actors & critic states
      sample actions $u_j^{a_s}$ & $u_j^{a_w}$
      compute rewards $r_j^{a_w}$ & $r_j^{a_s}$ for $u_j^{a_s}$ & $u_j^{a_w}$
      compute message m_j & value function V_j
      $j \leftarrow j + 1$
    compute global reward r^g
    for *j* = *t* to 1 **do**
      compute discounted reward $G_j^{a_s}$ and $G_j^{a_w}$
      estimate action-value functions $Q_j^{a_w}$ & $Q_j^{a_s}$
      compute advantages $A_j^{a_s}$ & $A_j^{a_w}$
      accumulate critic gradient $\Delta\theta^c$
      accumulate actor gradients $\Delta\theta^{a_s}$ &$\Delta\theta^{a_w}$
    update critic $\theta_{i+1}^c = \theta_i^c - \alpha\Delta\theta^c$
    update actors as $\theta_{i+1}^{a_s} = \theta_i^{a_s} + \alpha\Delta\theta^{a_s}$ & $\theta_{i+1}^{a_w} = \theta_i^{a_w} + \alpha\Delta\theta^{a_w}$
  return θ^{a_s}, θ^{a_w} & θ^m

and $A_j^{a_s}$ for actors as $Q_j^{a_w} - V_j$ and $Q_j^{a_s} - V_j$. At any step, one of the two ground-truth actions y_j^s / y_j^w is empty. Therefore, the computed value and action-value functions V_j and Q_j at that step intuitively become agent-specific, resulting in independent agent learning. Finally, agent specific advantage functions are used to compute actor gradients as $\nabla_{\theta^{a_w}} \log \pi_j^{a_w} A_j^{a_w}$ and $\nabla_{\theta^{a_s}} \log \pi_j^{a_s} A_j^{a_s}$. Importantly, value, action-value and advantage can be calculated in a single forward pass of the actor and critic for each agent. See appendix for details and proofs.

Communication. The communicator m (Figure 1, red circles) passes messages between the actors. Actor previous hidden states and contexts, $h_j^{a_s}$, $h_j^{a_w}$, c_j^s and c_j^w, are fed to m and a sigmoidal m_j is obtained. Value m_j is fed to a_s while $1 - m_j$ is fed to a_w. The gradient of m_j flows between actors during backpropagation and provides rich training signal that minimizes the learning effort.

See Algorithm 1 for DiMAC training algorithm details.

4 Experiments

4.1 Dataset

We preprocessed and filtered radiology reports from two medical centers in the USA (Courtesy of Princeton Radiology, Princeton and University of Colorado Health). [1] The resulting dataset comprises 37,408 radiology reports, which we randomly split into training (31,808), validation (3,740) and test sets (1,860). Table 3 gives dataset statistics.

	FINDINGS	IMPRESSIONS
#w per sentence	10.54 (06.53)	8.52 (05.80)
#s per report	8.23 (04.68)	1.75 (01.16)
#w per report	86.77 (64.72)	14.89 (15.81)

Table 3: Dataset statistics: number of words/sentences per sentence/report. Standard deviation in parentheses.

4.2 Experimental Setup

Training labels. Given an IMPRESSIONS sentence, we find a unique FINDINGS sentence with the highest sentence similarity score. We follow Chen and Bansal (2018) and Liu and Lapata (2019) and use ROUGE-L as the sentence similarity scorer. Furthermore, they use a greedy matching algorithm that takes similarity scores of all IMPRESSIONS and FINDINGS sentence combinations and yields a sequence of unique FINDINGS indices $\{y_1^s, \cdots\}$ of size equivalent to the length of IMPRESSIONS. There is a 1-to-1 correspondence between FINDINGS sentences at indices and IMPRESSIONS sentences. We refer to the papers for more details. These 1-to-1 correspondence are used for abstractor pretraining.

We use AutoPhrase (Shang et al., 2018) to extract keywords from training reports automatically. We select only high-quality keywords, K, and avoid too frequent ones as these can bias the system to only perform keyword selection. We implement an empirical threshold determined by hyperparameter search experiments.[2] We then find a subset of keywords, K^F, in FINDINGS F and compile their indices $\{y_1^w, \cdots\}$.

As the two extractors run for the same number of steps, we interleave the above sentence and word indices $\{y^s, \cdots\}$ and $\{y^w, \cdots\}$ into one sequence. In more detail, given a sentence index, all keywords indices within that sentence are placed in the sequence, followed by its index. A binary switch variable y^q (with values 0 and 1) distinguishes the

[1] Sentences split using Stanford CoreNLP (Manning et al., 2014). The following reports are excluded: (a) no FINDINGS and/or IMPRESSIONS; (b) FINDINGS has fewer than 3 words; (c) FINDINGS has fewer words or fewer sentences than IMPRESSIONS. We replace special tokens like numbers, dates and abbreviations and used scispacy lemmatization.

[2] AutoPhrase ranks keywords using a quality score based on frequency. The threshold is set on this score.

index type in the sequence, i.e., index refers to sentence vs. keyword. Both extractors require, during a decoding step j, training labels y_j^s and y_j^w; we set the value of "non-available type" as indicated by y_j^q to $\emptyset$. For example, when y_j^q is 0, y_j^w is $\emptyset$. Overall, an element in the final sequence is a tuple of y^q, y^s and y^w and provides training labels for the switch, word and sentence extractor networks. See Appendix A for details on the interleaving of indices.

Hyperparameters. Included in Appendix C.

Evaluation measure. We follow standard practice and evaluate the quality of generated IMPRESSIONS by comparing against ground-truth IMPRESSIONS using ROUGE (Lin, 2004).

4.3 Baseline Models

In this section we describe the baselines we compare our model against: a wide variety of extractive and abstractive systems.

Extractive systems

LexRank (Erkan and Radev, 2011) is a graph-based method for computing relative importance in extractive summarization.

Abstractive systems

PTGEN (See et al., 2017) introduces an encoder-decoder model that can copy words from the source text via pointing, while retaining the ability to produce novel words through the generator.

PTGEN+Coverage (See et al., 2017) introduces a coverage mechanism to the original PTGEN model to avoid repetition.

Zhang et al. (2018) provides an automatic generation system for radiology IMPRESSIONS using neural seq2seq learning. The model encodes background information of the radiology study and uses this information to guide the decoding process.

Self supervised learning has recently gained popularity as parameters of large models can be trained with little to no labeled data. Pre-trained language models in which a transformer encoder is trained to reconstruct the original text from masked text, e.g., BERT (Devlin et al., 2018), have become an important component in recent summarization models (Liu and Lapata, 2019; Zhang et al., 2020; Zaheer et al., 2020). **We also present results from experiments using these summarization models**. Additionally, we experimented with a **pre-trained seq2seq model** which is learned using different self supervised techniques to reconstruct the original text, e.g., BART (Lewis et al., 2019).

BertSumExtAbs (Liu and Lapata, 2019) is an encoder-decoder summarization framework that adopts BERT as its encoder. BERT is replaced by ClinicalBERT (Alsentzer et al., 2019) in all our experiments as it is adapted for the medical domain. At the first stage, a model with the BERT encoder accomplishes an extraction task. Then, the trained BERT encoder and a 6-layered transformer (Vaswani et al., 2017) are combined to form an abstractive system. As the encoder in the abstractive system is pre-trained multiple times in comparison to the decoder, two separate Adam optimizers (each with different warm-up steps and learning rates) are used during training. As the training is performed in two stages, **BertSumExtAbs** serves as the two-stage abstractive summarization system baseline for our experiments.[3] We also include results from **BERTSUMAbs**, a single-stage version in which encoder and decoder are trained only on the abstractive task.

BART (Lewis et al., 2019) is a state of the art transformer-based seq2seq model similar to BERTSUMAbs. However, unlike BERTSUMAbs's fine-tuning of the encoder and denovo training of the decoder, for BART, both encoder and decoder are only fine-tuned.

Sentence Rewrite (Chen and Bansal, 2018) is a two-step summarization model that initially extracts and then rewrites the sentences. This model serves as a two-step single agent baseline system for our experiments.

5 Results

In this section, we compare results from our model and various baselines using both automatic and human evaluation.

Automatic Evaluation. Table 4 shows report summarization results of various models trained and tested on the same data. Our DiMAC model surpasses extractive-only and abstractive-only baselines, including LexRank and PTGEN+Coverage. It also outperforms the two-step single agent baseline model (Sentence Rewrite (Chen and Bansal, 2018)) and the two-stage BERTSUMExtAbs (Liu and Lapata, 2019). Besides the pre-trained encoder

[3] We require hyperparameters somewhat different from the standard setup due to the small radiology report data size. Hyperparameter tuning yielded the following values. Batch size and initial learning rate of BERTSumExt are set to 16 and 5e-4, batch size in BERTSumExtAbs is 8 and initial learning rates of BERT and transformer decoder in BERTSumExtAbs are 0.0005 and 0.005.

Models	ROUGE-1	ROUGE-2	ROUGE-L
LexRank (Erkan and Radev, 2011)	27.33	14.78	29.8
PTGEN (See et al., 2017)	39.82	17.35	38.04
PTGEN+Coverage (See et al., 2017)	41.22	19.61	40.87
Zhang et al. (2018)	44.16	22.67	43.07
BERTSUMAbs (Liu and Lapata, 2019)	49.82	41.02	49.39
BERTSUMExtAbs (Liu and Lapata, 2019)	52.70	43.21	52.19
BART (Lewis et al., 2019)	41.23	29.02	40.02
Sentence Rewrite (Chen and Bansal, 2018)	59.82	48.54	59.11
DiMAC	**62.65**	**51.55**	**61.06**

Table 4: Results for baseline methods and DiMAC on the test split of the medical reports. The experimental setup is the same for all methods, i.e., the same train/validation/test split of the medical reports was used. Additionally, as DiMAC is a multi-agent two-step system built on top of Sentence Rewrite (Chen and Bansal, 2018) (a single-agent two-step setup), we keep abstractor and all hyperparameters except those specific to DiMAC the same for a fair comparison. All ROUGE scores have a 95% confidence interval of at most ±0.50 as calculated by the official ROUGE script.

of BertSumExtAbs, which is an advantage compared to other baselines, a denovo training of a large size decoder with a relatively small number of radiology reports may have led to overfitting. This might explain the scores compared to the two-step systems. Furthermore, a highly sophisticated semi-supervised training of the encoder and decoder of BART-base resulted in lower performance compared to our model, despite the relatively larger size (100x) of BART. We hypothesize that pre-training mostly on a different domain text (e.g., Wikipedia, Books Corpus and News) and fine-tuning on small data could have adversely affected BART's performance in our setting. The domain difference may also contribute to the relatively lower performance of BART-base versus BERTSUMExtAbs, thereby signifying the importance of pre-training with relevant domain text.

Moreover, DiMAC offers approximately 18 to 28% performance gains over (Zhang et al., 2018), a single-step single-agent summarization system designed specifically for the radiology domain. In our opinion, the performance improvements observed with DiMAC are likely driven by the extract-then-abstract mechanism combined with auxiliary (and salient) information from keywords, which mimics the actual reasoning process of radiologists.

It is important to note that our model supports user-level validation by linking the predicted IMPRESSIONS sentences to sentences in FINDINGS, making the results explainable to radiologists and referring physicians.

Human Evaluation. To assess the overall quality and factual correctness (Zhang et al., 2019) of the IMPRESSIONS generated by DiMAC, we obtained evaluations from two board-certified radiologists. We randomly selected 16 radiology reports from the test set. For each radiology report, we presented to the evaluators its FINDINGS and three (blinded) versions of the summary, i.e., IMPRESSIONS: (1) the ground truth, (2) Sentence Rewrite (Chen and Bansal, 2018) and (3) DiMAC. As Sentence Rewrite has a similar two-step approach, i.e., extract-then-abstract, we evaluate the qualitative performance of DiMAC with Sentence Rewrite as the base model (instead of BERTSUMExtAbs as it is a two-stage single-step system and also had lower Rouge scores compared to Sentence Rewrite).

	Win	Tie	Lose	Gwet AC1
DiMAC vs. Base model				
Overall quality	25.00	59.37	15.63	.305
Factual correctness	12.50	84.37	03.13	.711
DiMAC vs. Ground Truth				
Overall quality	25.00	46.87	28.13	.082
Factual correctness	21.87	53.13	25.00	-.080

Table 5: Percentage of 16 radiology reports for which human evaluators rated DiMAC better than (win), the same as (tie) or worse than (lose) the base model and ground truth on overall quality and factual correctness. We also provide Gwet's Agreement Coefficient as a measure of agreement between raters; values below 0.2 indicate poor agreement, values above 0.8 indicate very good agreement.

We shuffled the three summaries such that the order cannot be guessed. Each radiologist rated the summaries on two measures in relation to the FINDINGS: (1) overall quality and (2) factual correctness and completeness. For example, the phrase "pleural effusions" is a fact (or imaging finding); but the phrase "small bilateral pleural effusions" is a more precise description and should therefore have a better overall quality score. For each measure,

we asked the radiologists to score the summary as 1, 2 or 3 for bad, borderline or good. Then we combined the assigned scores under two comparisons: (1) our model versus the base model and (2) our model versus ground truth.

We have 32 evaluations in total: 2 radiologists $\times$ 16 reports. We compared the scores provided by the radiologists to determine if they were the same (tie), higher (win) or lower (lose) for our model vs. ground truth and our model vs. base model. Table 5 shows that DiMAC has clearly better factual correctness than the base model: 12.5% of cases are better, 3.13% are worse; gwet AC1 (Gwet, 2008) inter-rater agreement for this result is strong. DiMAC exceeds the base model in 25% (vs. 15.6% "lose") of evaluations for overall quality with moderate inter-rater agreement. DiMAC is only slightly worse than ground truth in overall quality (win: 25%, lose: 28.13%) and factual correctness (win: 21.87%, lose: 25%) – although inter-rater agreement is low in this case.

5.1 Qualitative Results Analysis

Table 6 shows a radiology report from our dataset (FINDINGS and IMPRESSIONS) and IMPRESSIONS generated by DiMAC and the base model. Due to the hierarchical connections between words and sentences, there is significant overlap between the extracted sentences and words. This phenomenon eventually contributes to the RL sentence extraction reward and helps to extract sentences with more keywords. The keywords include disease or clinical diagnoses (e.g., nodule, lymphadenopathy, effusion), anatomical concepts (e.g., hepatic) and qualifiers (e.g., recent, multiple, bilateral). The baseline model (Chen and Bansal, 2018) erroneously states "right greater than left pleural effusions", i.e., it hallucinates. In the sentence "There is no axillary or hilar lymphadenopathy", the sentence reward is low and eventually it is not extracted despite having the keyword "lymphadenopathy".

FINDINGS from the report from a medical site
There are **multiple bilateral** lung **nodules** , most consistent with metastatic disease .
There are more nodules on the right than the left .
An enlarged prevascular lymph node measures 0.6 x 0.4 cm .
There is no axillary or hilar **lymphadenopathy** .
No pleural or pericardial **effusion** is seen .
There is calcification in the aortic valve and coronary arteries .
There are numerous large **hepatic** masses which have been better described on **recent** ct scan of the abdomen .
There is degenerative disease in the thoracic spine with mild compression of the superior endplate of a lower thoracic vertebral body .
No suspicious osseous lesion is seen .
IMPRESSIONS from the report from a medical site
Multiple bilateral lung nodules , consistent with metastatic disease .
Mediastinal lymphadenopathy .
Multiple liver masses .
IMPRESSIONS generated by DiMAC
Multiple bilateral lung nodules , most consistent with metastatic disease .
No pleural effusions .
Numerous hepatic masses , better described on recent ct scan of the abdomen .
IMPRESSIONS generated by base model
Multiple bilateral lung nodules, most consistent with metastatic disease .
right greater than left pleural effusions .
enlarged right paratracheal lymph node .
numerous hepatic masses .

Table 6: FINDINGS and IMPRESSIONS of a radiology report from the report from a medical site and IMPRESSIONS generated by base model and DiMAC. Extracted sentences are highlighted in blue. Extracted words are shown in bold and underlined. The base model (Chen and Bansal, 2018) erroneously states "right greater than left pleural effusions", i.e., it hallucinates.

6 Related Works

Abstractive Summarization. An abstractive summary is a text consisting of novel phrases describing the content of the original text. Abstractive summarization involves a cascade of topic fusion and text generation (Hovy et al., 1999). Each task in this cascade typically requires expert-derived annotations, which is labor-intensive and time-consuming. Thus, many recent abstractive summarization approaches focus on supervised/semi-supervised single-step end-to-end trainable models that implicitly address the sub-tasks of content acquisition and paraphrasing.

As part of two-stage but single step abstractive summarization, a pretrained encoder first learns the extraction task independently. Then the pretrained encoder is embedded into an encoder-decoder abstractive summarization model to assist in better referencing the source content, e.g., Liu and Lapata (2019); Hsu et al. (2018). On the other hand, in two-step abstractive summarization, extractive summarization is followed by abstractive summarization and is trained end-to-end, e.g., Chen and Bansal (2018). Contrary to the two-stage single-step approach, both extractive and abstractive summarization are pretrained (and function) separately in a two-step approach; however, an RL-based end-

to-end training enables alignment between them to generate better summaries. DiMAC is a two-step abstractive system.

Multi-agent Reinforcement Learning (MARL). In a single-agent actor-critic (Sutton et al., 1999; Konda and Tsitsiklis, 2000) policy gradient method, an agent policy θ^{π} is optimized by following a gradient computed using a value function estimated by a critic. The simplest MARL setup applies policy gradients independently (each agent with its own actor and critic) and thereby restricts each agent to learn only from its own action history (Tan, 1993). From each agent's point of view in this setting, the environment is not stationary and therefore, the RL stationary environment assumption is violated.

MARL with communication or collaboration protocols. Foerster et al. (2018) proposed counterfactual policy gradients, which is an actor-critic policy gradient that leverages a centralized counterfactual critic that estimates value function for each actor by using actions performed by the other agents. However, unlike our setting, actors in (Foerster et al., 2018) are similar and share parameters. Additionally, the parameter sharing scheme has the limitation that the agents lack tighter coordination. Foerster et al. (2016), Sukhbaatar et al. (2016) and Mordatch and Abbeel (2018) proposed to tightly coordinate independent agents rather than use a dedicated channel. As incorporating an explicit communication channel mimics human (bidirectional) interactions, we design a similar Differentiable Multi-agent Actor-Critic (DiMAC) RL for our setup. In DiMAC, each agent selects one of its actions and communicates with the others at every point in time. Thus, the resulting joint action (influenced by the agents' communication) would aim to reach the desired (optimal) goal. In the future, we will experiment with more variations of MARL (such as counter-factual critic) and transformer-based networks.

7 Conclusion

In this work, we introduce a novel extractive approach into a two-step RL-based summarization task (extractive-then-abstractive). This approach is a MARL (rather than the traditional single-agent RL) which includes a new agent that extracts salient keywords from the source text and collaborates with an agent that extracts salient sentences. We also present a Differentiable Multi-agent Actor-Critic (DiMAC) learning method, a novel yet simple MARL training for independent agents communicating via a dedicated channel. We apply the proposed two-step summarization model with DiMAC MARL training to English radiology reports. Results from our experiments indicate, based on automatic and human expert evaluations, that the DiMAC summarization model can outperform existing baseline models for text summarization. Our summarization model generates the IMPRESSIONS to reflect human-level inference and actionable information (e.g., salient sentences and keywords) towards supporting improved workflow efficiency and better-informed clinical diagnosis based on medical imaging findings.

Acknowledgments

We thank Dr. Asik Ali Mohamed Ali and Dr. Abishek Balachandran for qualifying radiology reports and anonymized summaries for human evaluation. We also thank Jashwanth N B and Siemens Healthineers supercomputing team for training infrastructure. Furthermore, we thank the anonymous reviewers for their valuable feedback.

Disclaimer. The concepts and information presented in this paper are based on research results that are not commercially available. Future commercial availability cannot be guaranteed.

References

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686, Melbourne, Australia. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

N Reed Dunnick and Curtis P Langlotz. 2008. The radiology report of the future: a summary of the 2007 intersociety conference. *Journal of the American College of Radiology*, 5(5):626–629.

Günes Erkan and Dragomir R. Radev. 2011. Lexrank: Graph-based lexical centrality as salience in text summarization. *CoRR*, abs/1109.2128.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Jakob N Foerster, Yannis M Assael, Nando De Freitas, and Shimon Whiteson. 2016. Learning to communicate with deep multi-agent reinforcement learning. *arXiv preprint arXiv:1605.06676*.

Kilem Li Gwet. 2008. Computing inter-rater reliability and its variance in the presence of high agreement. *British Journal of Mathematical and Statistical Psychology*, 61(1):29–48.

Eduard Hovy, Chin-Yew Lin, et al. 1999. Automated text summarization in summarist. *Advances in automatic text summarization*, 14:81–94.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141. Association for Computational Linguistics.

Aishwarya Jadhav and Vaibhav Rajan. 2018. Extractive summarization with SWAP-NET: Sentences and words from alternating pointer networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 142–151, Melbourne, Australia. Association for Computational Linguistics.

Vijay R Konda and John N Tsitsiklis. 2000. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014. Citeseer.

Landon Kraemer and Bikramjit Banerjee. 2016. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190:82–94.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

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.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Igor Mordatch and Pieter Abbeel. 2018. Emergence of grounded compositional language in multi-agent populations. In *Thirty-second AAAI conference on artificial intelligence*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *IEEE Transactions on Knowledge and Data Engineering*, 30(10):1825–1837.

Sainbayar Sukhbaatar, Rob Fergus, et al. 2016. Learning multiagent communication with backpropagation. *Advances in neural information processing systems*, 29:2244–2252.

Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. 1999. Policy gradient methods for reinforcement learning with function approximation. In *NIPs*, volume 99, pages 1057–1063. Citeseer.

Ming Tan. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pages 330–337.

Zheng Tian, Shihao Zou, Ian Davies, Tim Warr, Lisheng Wu, Haitham Bou Ammar, and Jun Wang. 2020. Learning to communicate implicitly by actions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7261–7268.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

A Wallis and P McCoubrie. 2011. The radiology report—are we getting the message across? *Clinical radiology*, 66(11):1015–1022.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*.

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

Yuhao Zhang, Daisy Yi Ding, Tianpei Qian, Christopher D. Manning, and Curtis P. Langlotz. 2018. Learning to summarize radiology findings. In *Proceedings of the Ninth International Workshop on Health Text Mining and Information Analysis*, pages 204–213, Brussels, Belgium. Association for Computational Linguistics.

Yuhao Zhang, Derek Merck, Emily Bao Tsai, Christopher D. Manning, and Curtis P. Langlotz. 2019. Optimizing the factual correctness of a summary: A study of summarizing radiology reports.

Appendix

A Training Labels

In any training episode, we use Rouge-L and compute similarity scores between sentences in FINDINGS and IMPRESSIONS. Then, for each IMPRESSIONS sentence, we find the FINDINGS sentence that has the highest similarity score, and we compile its index. Furthermore, index compilation is a selection without replacement process, i.e., each sentence will only be selected once. This yields a sequence of unique sentence indices $\{y_1^s, \cdots\}$ of a size equivalent to the length of IMPRESSIONS. Additionally, we flatten FINDINGS sentences $\{s_1, \cdots, s_n\}$ to a long sequence of words $\{w_1, \cdots, w_m\}$. We then find words that are in the given keywords set K and compile their indices $\{y_1^w, \cdots\}$. For example, in Table 1, salient sentence and word indices are {1, 7, 8} and {6, 7, 9, 10, $\cdots$, 81, 82} respectively.

Finally, we interleave sentences and word indices $\{y_1^s, \cdots\}$ and $\{y_1^w, \cdots\}$ into one sequence to train extractors. Basically, given a sentence index, all keywords indices within that sentence are placed in the sequence. In addition, we use a binary switch variable y^q (with values 0 and 1) to distinguish the index type in the sequence, i.e., y^q=0 implies sentence index and y^q=1 implies word index. Thus, the length of the binary switch variables sequence is the same as the interleaved indices. As extractors run for the same number of steps, training requires the labels y_j^s and y_j^w at any step j. However, the interleave sequence at any step includes only one out of the two. So, we set the value of "non-available type" as indicated by y_j^q to $\emptyset$. Overall, an element in the final sequence is a tuple of y^q, y^s and y^w. For Table 1, the final sequence of training labels is {(1, $\emptyset$, 6), (1, $\emptyset$, 7), (1, $\emptyset$, 9), (1, $\emptyset$, 10), (0, 1, $\emptyset$), $\cdots$, (1, $\emptyset$, 81), (1, $\emptyset$, 82), (0, 8, $\emptyset$)}.

B Encoder-Extractor Training

Algorithm. 2 shows the training of word encoder (E_{w2w}), sentence convolutional network (*Conv*), sentence encoder (E_{s2s}), word extractor (D_{w2w}), sentence extractor (D_{s2s}) and switch network (*switch*).

Algorithm 2 Encoder-Extractor Training

procedure TRAIN-JOINT-EXTRACTORS
  Random Initialize: $E_{w2w}, Conv, E_{s2s}, D_{w2w}, D_{s2s}$ & *switch*
  for 1 to | *Reports* | **do**
    $\{h_1, \cdots, h_n\} \leftarrow Conv(\{s_1, \cdots, s_n\})$
    $\{h_1^{E_{s2s}}, \cdots\} \leftarrow E_{s2s}(\{h_1, \cdots\})$
    $\{h_1^{E_{w2w}}, \cdots, \} \leftarrow E_{w2w}(\{w_1, \cdots\})$
    $Loss \leftarrow Array()$
    $h_1^{D_{w2w}}, h_1^{D_{s2s}} \leftarrow h_m^{E_{w2w}}, h_n^{E_{s2s}}$
    for j = 1 to t **do**
      $\boldsymbol{\alpha}^w \leftarrow Attn(h_j^{D_{w2w}}, \{h_1^{E_{w2w}}, \dots\})$
      $c_j^w \leftarrow \sum_{i=1}^{m} \alpha_i^w \times h_i^{E_{w2w}}$
      $\boldsymbol{\alpha}^s \leftarrow Attn(h_j^{D_{s2s}}, \{h_1^{D_{s2s}}, \dots\})$
      $c_j^s \leftarrow \sum_{k=1}^{n} \alpha_k^s \times h_k^{E_{s2s}}$
      $q_j \leftarrow switch(h_j^{D_{w2w}}, c_j^w, h_j^{D_{s2s}}, c_j^s))$
      $h_{j+1}^{D_{w2w}} \leftarrow D_{w2w}(h_j^{D_{w2w}}, c_j^w, c_j^s)$
      $h_{j+1}^{D_{s2s}} \leftarrow D_{s2s}(h_{j-1}^{D_{s2s}}, c_j^w, c_j^s)$
      $Loss$.ADD$(-(1 - y_j^q)(y_j^w \log \boldsymbol{\alpha}^w))$
      $Loss$.ADD$(-y_j^q(y_j^s \log \boldsymbol{\alpha}^s))$
      $Loss$.ADD$(-y_j^q \log q_j)$
    compute gradients, $\{\Delta_{E_{w2w}} Loss, \cdots\}$
    update $E_{w2w}, Conv, E_{s2s}, D_{w2s}, D_{s2s}$ & *switch*
  return $E_{w2w}, Conv, E_{s2s}, D_{w2s}, D_{s2s}$ & *switch*

C Hyperparameter

We set the maximum limit for words in a report to 800 tokens, and the maximum number of sentences is truncated to 60 per report. We use word2vec (Mikolov et al., 2013) on the training set to generate word embeddings of 128 dimensions. The vocabulary is 50,000 most common words in the training set. The dimension of each intermediate sentence representation is 300 after using 1-D convolution filters with 3 different windows sizes (i.e. 3, 4, and 5). The dimension of all the LSTMs in

our framework is 256. The optimizer used is Adam with a learning rate of 0.001 in the pre-training phase and 0.0001 in the RL training phase. We apply gradient clipping to alleviate gradient explosion using a 2-norm of 1.5. We adopt the early stopping method on the validation set. In the RL setting, the discounted factor γ is set as 0.95. At test time, we use beam size 5 for beam search.

D Single Agent Actor-Critic

In the case of a single agent actor-critic RL, for any training episode, actor a uses its policy network π^a and samples actions $\{u_1^a, \cdots, u_t^a\}$ for t time steps with each action u_k^a receiving a reward r_j^a. Furthermore, at step j, a discounted reward is computed as $G_j^a = \sum_{l=0}^{t-j} \gamma^l r_{j+l}^a$.

A batch of training episodes is used to estimate the actor's action value at step j as $Q_j^a = \mathbb{E}_{u_{j:t}^a, h_{j:t}^a}[G_j^a \mid h_j^a, u_j^a]$. Similarly, the critic (c) estimates a value function for step j as $V_j = \mathbb{E}_{h_j^a}[G_j^a \mid h_j^a]$. An advantage function is computed as $A_j^a = Q_j^a - V_j$. Policy gradient theorem computes the gradient to update the actor parameter θ^a as

$$\Delta\theta^a = \mathbb{E}_{\theta^a}\left[\sum_{j=1}^{t} \nabla_{\theta^a} \log \pi_j^a A_j^a\right] \quad (1)$$

The value function component V_j in the policy gradient helps to reduce the variance without changing the expectation as

$$\begin{aligned} &-\mathbb{E}_{\theta^a}\left[\nabla_{\theta^a} \log \pi_j^a V_j\right] \\ &= -\sum_h d^{\pi_j^a}(h) \sum_{u^a} \nabla_{\theta^a} \pi_j^a V_j \\ &= -\sum_h d^{\pi_j^a}(h) V_j \nabla_{\theta^a} \sum_{u^a} \pi_j^a \\ &= 0 \end{aligned}$$

where $d^{\pi_j^a}(h)$ is the discounted ergodic state distribution (Sutton et al., 1999). V_j is a function of state and not action, thus moved outside ∇, and since $\sum_{u^a} \pi_j^a$=1, the gradient becomes 0. $\Delta\theta^a$ is empirically estimated using N episodes in a training batch as

$$\Delta\theta^a \approx \frac{1}{N} \sum_{i=1}^{N} \left[\sum_{j=1}^{t} \nabla_{\theta^a} \log \pi_j^a A_j^a\right] \quad (2)$$

E Multi Agent Actor-Critic

In the case of multi-agent actor-critic RL with a set of actors, $\boldsymbol{a}$=$\{\cdots, a_k, \cdots\}$, for any training episode, an actor a_k uses its policy network π^{a_k} and samples actions $\{u_1^{a_k}, \cdots, u_t^{a_k}\}$ for t time steps with each action $u_j^{a_k}$ receiving a reward $r_j^{a_k}$. Furthermore, at step j, a discounted reward for a_k is computed as $G_j^{a_k} = \sum_{l=0}^{t-j} \gamma^l r_{j+l}^{a_k}$.

Like the single agent actor-critic, a batch of training episodes is used to estimate the action value of a_k at step j as $Q_j^{a_k} = \mathbb{E}_{u_{j:t}^{a_k}, h_{j:t}^{a_k}}[G_j^{a_k} \mid h_j^{a_k}, u_j^{a_k}]$.

The contribution of value function from a centralized critic at any step j in the overall gradient is computed as

$$-\mathbb{E}_{\boldsymbol{\theta^a}}\left[\nabla_{\boldsymbol{\theta^a}} \log \boldsymbol{\pi_j^a} V_j\right]$$

where $\boldsymbol{\theta^a}$ and $\boldsymbol{\pi^a}$ are the actors' $\boldsymbol{a}$ joint parameters and policies respectively. V_j is the value function computed by the critic at step j. We drop the step notation j subsequently as all notations are specific to step j. The agent-wise break of policies and the contribution of the value function in the overall gradient is

$$\begin{aligned} &= -\sum_h d^{\boldsymbol{\pi^a}}(h) \sum_{a_k} \sum_{u^{a_k}} \nabla_{\theta^{a^k}} \pi^{a_k} V \\ &= 0 \end{aligned}$$

where $d^{\boldsymbol{\pi^a}}$ is the discounted ergodic state distribution, u^{a_k} is agent a_k action and V is the estimated value function by the critic. Although two actors are running at each step in our DiMAC training, only one of them is active while the other is on pause ($\emptyset$ selection). Therefore, the contribution of the term $\sum_{\boldsymbol{a}} \sum_{u^{a_k}} \nabla_{\theta^{a^k}} \pi^{a_k} V$ is similar to a single-agent scenario, and therefore, the gradient is 0. Furthermore, the critic estimated value ensures that the active agent gets rewarded for its action leading to the overall success.