事前学習済Sequence-to-Sequenceモデルと重要度モデルの結合による生成型要約

斉藤 いつみ　西田 京介　西田 光甫　浅野 久子　富田 準二
日本電信電話株式会社 NTT メディアインテリジェンス研究所

1 はじめに

BART [5] や T5 [10] など Encoder-Decoder 型の事前学習 Sequence-to-Sequence（Seq-to-Seq）モデルが提案され，生成タスクにおいても事前学習が有効であることが示されている．本研究で扱う生成型要約タスクにおいても，現在の SOTA は BART を fine-tuning したモデルであり，事前学習を用いないモデルに比べ大幅に精度が向上している．

一方，要約においては「ソーステキストのどの部分が重要か？」を正しく予測することが本質的には重要であり，事前学習を用いない従来手法では，抽出型の要約と生成型の要約を組み合わせるモデルが複数提案されている [3]．BART などのモデルはこれらの従来手法の精度を上回っているが,「どの部分が重要か」のモデル化まで事前学習を通して獲得できているのか否かについては明らかにされていない．

本研究では事前学習 Seq-to-Seq モデルの能力を明らかにするとともに重要箇所予測の効果を検証するため，事前学習 Seq-to-Seq モデルとソーステキストの重要な単語を予測する重要度モデルの結合方法について検証した．主な貢献は以下の 3 点である．

- Seq-to-Seq モデルと重要度モデルの結合モデルとして，重要度モデルの予測値に基づいて抽出したトークン列を追加のテキスト情報として Seq-to-Seq モデルに明示的に与える **CIT** (Conditional summarization model with Important Tokens) モデルを提案した．
- BART と重要度モデルの結合モデルについて 9 種の組合せを CNN/DM [4]，XSum [8] データセットにて検証した．CIT と Selective Encoding [12] の組合せが両データセットにおいて ROUGE 値にて BART を上回る最高精度を達成した．
- CIT モデルでは，抽出トークン数と要約トークン数を近づけるように学習可能なため，抽出トークン数を変更することで要約長を制御可能であることを示した．

2 問題定義

本研究で取り組む 2 タスクを定式化する．メインタスクは生成要約，サブタスクが重要トークン予測である．

問題 1 (生成要約). ソーステキスト $X = \{x_i, x_2, ...x_L\}$ を入力とし，要約テキスト $Y = \{y_i, y_2, ...y_T\}$ を出力する．X, Y はそれぞれトークン列とする．

問題 2 (重要トークン予測). ソーステキスト $X = \{x_i, x_2, ...x_L\}$ を入力とし，ソーステキスト中の各トークンに対応する重要度（saliency） スコア $S = \{S_1, S_2, ...S_L\}$ を出力する．$0 \leq S_l \leq 1$ とする．

3 事前学習 Seq-to-Seq モデル

本章では現在最も精度が高い BART [5] について説明する．BART の構造は Transformer 型の Encoder-Decoder [11] であり，ノイズを入れたテキストを Encoder でエンコードし Decoder で復元を行う事前学習を行う．モデルは以下のように定義される．

Encoder Encoder は，M 層の Transformer Encoder ブロックからなる．入力 $X = \{x_i, x_2, ...x_L\}$ を受け付け，M 層のブロックを作用させた表現

$$H_e^M = \{h_{e1}^M, h_{e2}^M, ...h_{eL}^M\} \in \mathbb{R}^{L \times d} \tag{1}$$

を得る．ここで，本ブロックは self-attention と 2 層のフィードフォワードネットワーク（FFN）からなる．

Decoder Decoder は M 層の Transformer Decoder ブロックからなる．Encoder の出力 H_e^M とモデルが 1 ステップ前までに出力した系列 $\{y_1, ..., y_{t-1}\}$ を入力とし，M 層のブロックを作用させた表現

$$H_d^M = \{h_{d1}^M, ..., h_{dt}^M\} \in \mathbb{R}^{t \times d} \tag{2}$$

を得る．各ステップ t において，h_{dt}^M に対し語彙サイズ V 次元への線形変換を行い，確率が最大となる y_t を次のトークンとして出力する．本ブロックは，self-attention，context-attention, 2 層の FFN からなる．

Multi-head Attention Transformer ブロックにおけるアテンション処理はすべて Multi-head Attention を用いる [11]．本処理は k 個のアテンションヘッドの結合からなり，$\mathrm{Multihead}(Q, K, V) = \mathrm{Concat}(\mathrm{head}_1, ..., \mathrm{head}_k)W^o$ と表される．各ヘッドは $\mathrm{head}_i = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$ である．ここで，Encoder-Decoder の第 m 層の self-attention では Q, K, V に同じベクトル表現 $H_\cdot^m$ を与え，Decoder の context-attention においては，Q に H_d^m，K と V に H_e^M をそれぞれ与える．

各ヘッドのアテンション $\mathrm{Attention}(\tilde{Q}, \tilde{K}, \tilde{V}) = A\tilde{V}$ における重み行列 A は式 (3) にて表される．

$$A = \mathrm{softmax}\left(\frac{\tilde{Q}\tilde{K}^\top}{\sqrt{d_k}}\right) \in \mathbb{R}^{I \times J} \tag{3}$$

ここで，$d_k = d/k$，$\tilde{Q} \in \mathbb{R}^{I \times d}$，$\tilde{K}, \tilde{V} \in \mathbb{R}^{J \times d}$ である．

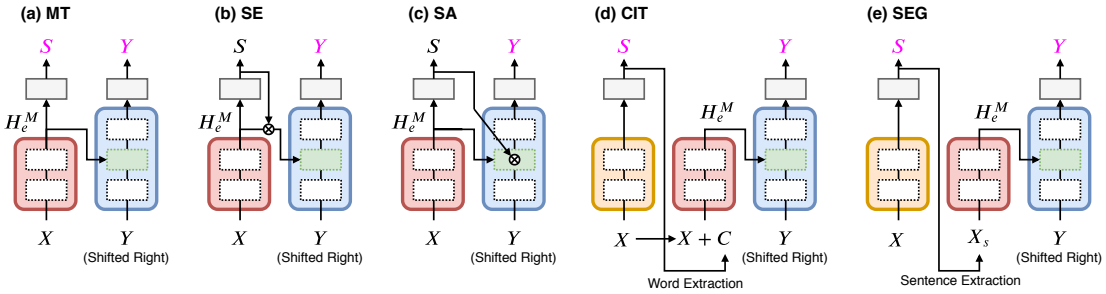

図 1: Seq-to-Seq モデル（赤ブロック: M 層の Transformer Encoder，青ブロック: M 層の Transformer Decoder）と重要度モデル（赤 or 黄ブロック: M 層の Transformer $\text{Encoder}_{\text{sal}}$）の結合方法．灰ブロックは線形変換，緑ブロックは context-attention を表す．(a-c) では Encoder と $\text{Encoder}_{\text{sal}}$ を共有する．(d-e) は重要度モデルを個別に学習する．X はソーステキスト，Y は要約テキスト，S は重要度スコア，C は重要トークン列，X_s は重要文集合，H_e^M は M 層の Encoder 出力を表す．紫色の出力は正解データを与えて学習する．

4 重要度モデル

本研究では Seq-to-Seq モデルに重要度モデルを結合する．重要度モデルは M 層の Transformer Encoder ブロック ($\text{Encoder}_{\text{sal}}$) と 1 層の線形変換層からなる．

4.1 重要度スコア

ソーステキスト中の第 l トークンの重要度スコア S_l を次のように定義する．$(1 \leq l \leq L)$

$$S_l = \sigma(W_1^\top \text{Encoder}_{\text{sal}}(X)_l + b_1) \tag{4}$$

ここで，$\text{Encoder}_{\text{sal}}()$ は $\text{Encoder}_{\text{sal}}$ の最終層の出力ベクトルを表す．$W_1 \in \mathbb{R}^d$ と b_1 は学習可能なパラメータであり，σ はシグモイド関数を表す．

4.2 重要度疑似正解の作成

重要度モデルはソーステキストの各トークン位置 l に対して予測 S_l の出力を行うものであるため，疑似的に正解を与えることで教師あり学習が可能になる．通常，要約データの正解は，ソーステキストと要約テキストのペアのみであるため，ソーステキストの各トークンに 1/0 の正解は与えられていない．しかし，ソース文書と要約文書の双方に含まれている単語が重要な単語である，とみなして両者のトークン系列のアライメントをとることで重要トークンの正解を疑似的に作成できる [3]．本研究においては，疑似正解を用いる場合と用いない場合のそれぞれの結合モデルを検証する．

5 結合モデル

提案モデル **CIT** を含む，重要度モデルを事前学習 Seq-to-Seq モデルに結合する複数の方式について，重要度モデルの正解の利用有無で分類して説明する．CIT, SEG のみ重要度モデルと Encoder-Decoder モデルを独立に学習し，それ以外のモデルは両者を同時に学習する．図 1 にそれぞれのモデルのイメージ図を示す．

5.1 重要度正解を与えないモデル

Selective Encoding (SE) 本モデルは，重要度スコア S_l を用いて Encoder にバイアスをかける [12]．具体的には，Encoder の最終出力 h_{el}^M に対して下記のように重み付けを行う．

$$\tilde{h}_{el}^M = h_{el}^M S_l \tag{5}$$

本モデルでは，Decoder に入力する h_{el}^M を $\tilde{h}_{el}^M$ で置き換える．文献 [12] では BiGRU を用いているが，本研究では公平な比較のため Transformer を用いて実装する．ここで，$\text{Encoder}_{\text{sal}}$ は Seq-to-Seq モデルの Encoder のパラメータを共有する．

Selective Attention (SA) 本モデルは，SE モデルと異なり Decoder 側のアテンションを重み付けする．具体的には，context-attention の第 i ヘッドの重み行列 $A_i \in \mathbb{R}^{T \times L}$ の各ステップ t のアテンション確率 $a_i^t \in \mathbb{R}^L$（A_i の第 t 行）を S_l により重み付けする．

$$\tilde{a}_{il}^t = \frac{a_{il}^t S_l}{\sum_l a_{il}^t S_l} \tag{6}$$

Gehrmann らが提案した類似手法では pointer-generator のコピー確率を重み付けしている [3]．一方，事前学習 Seq-to-Seq モデルはコピー機構を有さないため，本研究では Decoder の全ての層の context-attention 計算において，上記の重み付けされたアテンション確率を用いて計算を行う．$\text{Encoder}_{\text{sal}}$ は Seq-to-Seq モデルの Encoder のパラメータを共有する．

5.2 重要度正解を与えるモデル

Multi-Task (MT) 重要度スコア S_l に対する正解データを追加利用し，重要度モデルと Seq-to-Seq モデルを同時に学習する．$\text{Encoder}_{\text{sal}}$ は Seq-to-Seq モデルの Encoder のパラメータを共有する．

SE + MT 本モデルは，SE モデルの学習において重要度スコア S_l に対する正解データを追加利用して要約との同時学習を行う．

SA + MT 本モデルは，SA モデルの学習において重要度スコア S_l に対する正解データを追加利用して要約との同時学習を行う．

CIT 本研究では，Seq-to-Seq モデルと重要度モデルの新たな結合モデル CIT を提案する．SE や SA モデルでは内部的に全てのトークンに重要度スコアの重み付けをするため，明示的に重要なトークンが選択されない．これに対して，本モデルでは，重要度モデル

と Seq-to-Seq モデルを独立に学習し，重要度モデルによって抽出された重要なトークン列を Seq-to-Seq モデルに対して追加のテキスト情報として与える．具体的には，ソーステキスト X を $\text{Encoder}_{\text{sal}}$ に入力して得られた重要度スコア S の高い順に K 個のトークン $C = \{c_1, ..., c_k\}$ を抽出する．このとき，C の順序はソーステキスト X の順序を保持する．そして，入力テキストを $\tilde{X} = \text{Concat(X, C)}$ として Seq-to-Seq モデルに与える．X と C の間には区切りを表す特殊トークンを挿入する．ここで，本モデルでは $\text{Encoder}_{\text{sal}}$ として RoBERTa [6] を初期値として用いる．

CIT + SE　CIT の Encoder において，SE により結合テキストに対して重要度スコア $S \in \mathbb{R}^{L+K}$ を教師無しで学習し $H_e^M \in \mathbb{R}^{L+K}$ に重み付けを行う．

CIT + SA　CIT+SE と同様に $S \in \mathbb{R}^{L+K}$ を教師無し学習し，$a_i^t \in \mathbb{R}^{L+K}$ に重み付けを行う．

Sentence Extraction then Generation (SEG) 本モデルは，個別に学習した $\text{Encoder}_{\text{sal}}$ から得たトークンレベルの重要度スコア S_l を用いて文レベルの重要度スコア S_j を計算し，上位 P 文を結合したものを入力 X_s として Seq-to-Seq で生成を行う．$S_j = \frac{1}{N_j} \sum_{l:x_l \in X_j} S_l$ と計算した．ここで N_j，X_j はそれぞれ j 番目の文に含まれるトークン数，トークン集合を表す．また，学習時はソーステキスト X と正解要約テキスト Y_{ref} の文同士の ROUGE-L が最大になる文集合 X_{ref} を入力として学習した．テスト時は，学習時の正解文数の平均を P として用いた．

5.3 損失関数

損失関数 L の最小化にあたり，5.1 節の重要度正解を与えないモデルでは，$L = L_{\text{encdec}}$ とし，要約モデルの損失関数 L_{encdec} のみを用いる．5.2 節の重要度正解を与えるモデルでは，$L = L_{\text{encdec}} + L_{\text{sal}}$ として重要度モデルの損失関数 L_{sal} を併せて最小化する．

要約モデル　cross entropy を用いて次のように定義する．N はデータの個数を表す．

$$L_{\text{encdec}} = -\frac{1}{NT} \sum_{n=1}^{N} \sum_{t=1}^{T} \log P(y_t^n) \tag{7}$$

重要度モデル　重要度モデルは，binary cross entropy を用いて次のように定義する．

$$L_{\text{sal}} = -\frac{1}{NL} \sum_{n=1}^{N} \sum_{l=1}^{L} \left\{ \begin{array}{l} r_l^n \log S_l^n + \\ (1 - r_l^n) \log(1 - S_l^n) \end{array} \right\} \tag{8}$$

ここで，r_l^n は n 番目のデータにおけるソーステキストの位置 l の重要度正解を表す．

6 評価実験

6.1 実験設定

データセット　要約データセットにおいて代表的な CNN/DM [4] と XSum [8] を用いた．CNN/DM は 3 文程度抽出率が高い要約データであり，XSum は 1 文程度の短く抽出率が低い要約データである．評価は，要約自動評価で標準的に用いられる ROUGE 値を用いて評価した．各データの概要を表 1 に示す．要約平均長は，各データの dev セットを fairseq[1] の byte-level BPE [9] にてサブワードに分割した単位で計算した．

set	train	dev	eval	要約平均トークン数
CNN/DM	287,227	13,368	11,490	69.6
XSum	203,150	11,279	11,267	26.6

表 1: 要約データ数と平均長

models	R1	R2	RL
Presum [7]	42.13	19.60	39.18
UniLM [2]	43.33	20.21	40.51
T5 [10]	43.52	21.55	40.69
BART [5]	44.16	21.28	40.90
BART (our fine-tuning)	43.79	21.00	40.58
MT	44.58	21.46	41.32
SE	44.59	21.49	41.28
SE + MT	44.77	21.50	41.51
SA	44.72	21.59	41.40
SA + MT	44.79	21.69	41.47
CIT	45.05	22.02	41.78
CIT + SE	**45.34**	**22.13**	**42.15**
CIT + SA	45.20	22.12	41.95
SEG	44.62	21.51	41.29

表 2: CNN/DM における要約評価結果．“our fine-tuning” は我々による Fine-tuning 結果（ベースライン）を表す．太字は全体の最高精度を表す．

models	R1	R2	RL
Lead3	16.30	1.60	11.95
Presum [7]	38.31	16.50	31.27
BART [5]	45.14	22.27	**37.25**
BART (our fine-tuning)	45.03	21.69	36.53
MT	44.68	21.43	36.12
SE	45.35	22.00	36.81
SE + MT	43.97	20.64	35.48
SA	45.34	22.02	36.82
SA + MT	44.67	21.41	36.15
CIT	45.33	21.92	36.68
CIT + SE	**45.71**	**22.30**	36.99
CIT + SA	45.37	21.94	36.75
SEG	41.03	18.20	32.75

表 3: XSum における要約評価結果．下線はベースラインモデルの改善の中で最高精度を表す．

学習用設定． fairseq を用いて各モデルの実装を行った．7 枚の NVIDIA V100 32GB GPU により学習を行った．CNN/DM の学習では，文献 [5] と同じ設定を用いた．XSum の学習は著者らに確認を取り，CNN/DM の設定からミニバッチの勾配累積に関するパラメータ UPDATE_FREQ を 2 に変更した．また，学習時における CIT の重要度モデルの抽出トークン数 K は，dev セットの精度を確認し，CNN/DM では正解要約長を 5 単位で bin 分割した値，XSum では固定長 30 と決定した．なお，評価時は CNN/DM ではの dev セットの平均要約長を K とし，XSum では学習時と同じ 30 とした．また，XSum では，重要トークン列に重複を含まないように設定した．

6.2 実験結果

重要度モデルを組み合わせることによって要約精度は向上するか？　表 2，表 3 に示す通り，両方のデータセットにおいて CIT + SE が最も良い結果となった．まず，CIT 単体でも精度が向上したことから，重要トークンの結合が重要な情報を Seq-to-Seq モデルに与える方法として優れていることがわかる．そして，CIT を SE や SA モデルと組み合わせることでさらなる精度の向上が確認できたことから，重要トークンの

[1] https://github.com/pytorch/fairseq

	CNN/DM		
models	R1	R2	RL
Lead3	40.3	17.7	36.6
Presum [7]	43.25	20.24	**39.63**
top$K$ tokens	**46.72**	**20.53**	37.73
top3 sentences	42.64	20.05	38.89
	XSum		
models	R1	R2	RL
Lead3	16.30	1.61	11.95
top$K$ tokens	24.60	3.17	15.15
top3 sentences	21.98	4.09	16.95

表 4: CNN/DM, XSum における重要度モデル評価結果. (CIT, SEG で用いた重要度単体モデル)

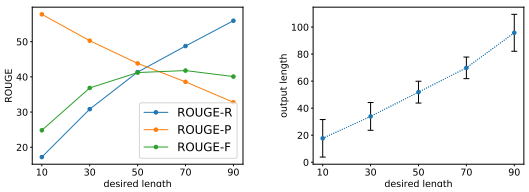

図 2: CIT の K を変化させた場合の CNN/DM における結果. a; 左) ROUGE-L recall, precision and F 値. b; 右): 平均出力長. エラーバーは標準誤差を表す.

中にはノイズも含まれるが，ソフトな重み付けを組み合わせることによって改善されていると考える．

SE，SA はいずれのデータでも精度が向上した．重要度正解を同時学習する MT，SE + MT，SA + MT は CNN/DM では精度が向上したが，XSum では精度が低下した．これは重要度の疑似正解の質の影響と考える．CNN/DM は要約が比較的長く抽出要素が強いため，疑似正解を作成する際にトークン同士のアライメントが取り易い．一方で，XSum データは要約が短く，ソーステキスト以外の表現で記述されることが多いため，アライメントが取り難く，疑似正解がノイズとして働いた．CIT においても，この影響により XSum における精度の向上幅は CNN/DM よりも狭いが，性質の異なるデータでいずれも最も高い性能を示していることから，頑健に動作する手法と言える．

重要度モデル単体のトークン抽出精度はどの程度か？ 表 4 に CIT の重要度スコア S_l によって上位に選択されたトークン列と要約テキストの ROUGE 値を評価した結果を示す．CNN/DM においては，重要トークンの特定が適切に行われていることがわかる．抽出型要約の従来手法の SOTA である Presum [7] と比較しても，R1 や R2 で高い精度を出しており，トークンレベルで重要な要素を抽出できていることがわかる．SE モデルや SA モデルの内部的に学習された S では各トークンの重要度に大きな差がなかったが，本モデルでは重要なトークンを明示的に考慮することができる．一方で，XSum は抽出率の低いデータであるため，全体的に重要度モデルの精度が低い．このことが要約精度の向上が CNN/DM よりも低くなっている原因と考える．今回検証した疑似的正解を与える手法は抽出率の高いデータにおいて特に効果を発揮したが，抽出率の低いデータにおいても重要度モデルの精度を向上させることでさらに精度向上が期待できる．

CIT モデルで重要トークン数 K を変化させた場合の出力はどうなるか？ CIT モデルの学習時の K に正解要約長を与えて学習するモデルについては，学習時に重要トークン数 K と要約トークン数 T に高い相関があるため，テスト時に K を変化させるだけで，指定した K と相関の高い要約長の出力を得ることができる．図 2-a,b) に CNN/DM データでの結果を示した．図 2-b) より，出力長の分散については学習データ数と関係があり，CNN/DM の要約トークン数の平均である 69.6 に近い $K = 70$ で最も出力要約長の分散が小さく，良い制御が出来ている．また，図 2-a) より，各長さで高い精度を維持していることがわかる．K を変化させて生成した例を表 5 に示す．

[K = 10 (重要語)] ban, beef, to, police, Cows, considered, holy, by, Hindu
[K = 10 (生成結果)] Cows are considered holy and revered by majority Hindu population .
[K = 30 (重要語)] in, Malegaon, Maharashtra, residents, a, ban, on, and, beef, to, mugshot, ', their, police, cow, slaughter, Cows, are, considered, holy, and, revered, by, majority, Hindu, population
[K = 30 (生成結果)] Police in Indian city of Malegaon, western state of Maharashtra, ask residents to take 'mugshot' of cattle . Cows considered holy and revered by majority Hindu population .
[参照要約] Authorities in the Indian city of Malegaon have asked residents to take a ' mugshot ' of their cattle . Cows are revered by the majority Hindu population , and many parts of the country have laws banning the slaughter of cattle . Officials in Malegaon believe this is the best way to solve cow slaughter cases and enforce the law .

表 5: CIT の出力長制御例．`https://edition.cnn.com/2015/04/02/asia/india-cow-mugshots/index.html` にてソーステキストを確認可能．K はサブワード単位で指定したが，表示は単語単位に修正．

7 おわりに

関連研究と議論 事前学習 Seq-to-Seq モデルにおいて重要度モデルを結合する効果の検証はこれまで行われていない．Presum [7] が最も本研究に類似したもので，BERT を重要文抽出タスクで fine-tuning したものを Encoder の初期値として用いるが，要約の学習時は重要スコア・トークンを考慮しない点で本研究と異なる．また従来の重要度結合モデル（[12, 3] 等）は要約精度の改善目的で利用されてきたが，長さ制御に有効であるとの報告はこれまで行われていない．

本研究の重要性 本研究は CNN/DM，XSum にて SOTA の精度を達成するとともに，複数方式の網羅的な実験により事前学習 Seq-to-Seq モデルに対して重要度モデルを結合することの意義を明らかにした点で，生成型要約研究の今後の方針について一定の指針を与えるものである．さらに，重要度モデルの出力をトークンレベルで明示的に生成モデルに組み込むことで要約長を自然に制御できることを明らかにした点で，産業上重要な課題である，要約率を指定した文書要約研究に対する貢献が大きいと考える．

参考文献

[1] J. Devlin et al. In *NAACL*, pages 4171–4186, 2019.
[2] L. Dong et al. In *NeurIPS*, pages 13042–13054. 2019.
[3] S. Gehrmann et al. In *EMNLP*, pages 4098–4109, 2018.
[4] K. M. Hermann et al. In *NIPS*, pages 1693–1701, 2015.
[5] M. Lewis et al. *arXiv*, 1910.13461, 2019.
[6] Y. Liu et al. *arXiv*, 1907.11692, 2019.
[7] Y. Liu et al. In *EMNLP-IJCNLP*, pages 3728–3738, 2019.
[8] S. Narayan et al. In *EMNLP*, pages 1797–1807, 2018.
[9] A. Radford et al. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.
[10] C. Raffel et al. *arXiv*, 1910.10683, 2019.
[11] A. Vaswani et al. In *NIPS*, pages 5998–6008, 2017.
[12] Q. Zhou et al. In *ACL*, pages 1095–1104, 2017.