Pointer-Generatorモデルにおけるデータ拡張に関する研究

大内　智仁 田伏　正佳
京都府立大学大学院生命環境科学研究科
t-ouchi@mei.kpu.ac.jp, tabuse@kpu.ac.jp

1 はじめに

近年、全世界のネットの情報量は、指数関数的に増加しており、2020年には、44ZBになると予想されている[1]。そのなかで、情報を効率よく取捨選択するには、自動要約の技術が必要不可欠となってくる。そこで、我々は、まず、ネットの情報のなかで、大部分を占めている英語の記事を日本語の要約に変換する研究を始めようとした。しかし、そのような英語の記事と日本語の要約がセットとなっているコーパスは存在しない。そのようにタスクによっては、自動要約システムで使えるコーパスは整備されていない。そこで、我々は、画像処理でよく使われるデータ拡張の手法を自然言語処理に応用できないかと考えた。

自然言語処理にデータ拡張を適用した研究がいくつか行われている。中間ベクトルにノイズを加える方法[2]や方言などで語尾を変化させてデータ拡張する方法[3]などである。しかし、いずれも文書分類に適応したもので、自動要約に適応したものはまだない。自動要約の性質として、重要でない部分はなくても生成する要約には影響が少ないという点がある。一方で、画像処理において、背景を変えてデータ拡張することがある。これは、背景は、対象とするタスクには直接影響がないことを前提にしている。よって我々は、画像処理のデータ拡張のひとつである背景の変更に着目して、自動要約における元記事の重要でない文の削除によるデータ拡張を行った。

自動要約には大別して2つある。1つは、抽出型要約で、入力記事の単語やフレーズまたは、文をそのまま抽出して組み合わせて要約を作るというものである。もう1つは、生成型要約で、入力記事をいったん中間ベクトルに直してから、その中間ベクトルをもとに、要約を生成するというものである。抽出型要約の問題点として、抜き出した単語やフレーズ、文をどう並べて繋げるのかという問題点がある。また、抽出された要約において、指示語が何をさしているのかが分からない要約になる可能性もある。それに対して、生成型要約では、一から生成するので、学習がうまくいけば、ひとつのまとまった要約になる可能性がある。文法も学習するので、抽出型要約のように、単語やフレーズをどう繋げていいかわからないということもない。ただし生成型要約は、入力記事と生成要約の大量のペアで学習する必要があり、コーパスを用意するのに手間がかかってしまう。

そこで本研究では、生成型要約において、データ拡張が効果あるかどうかを検証する。データ拡張の提案手法として、入力記事において重要度の一番低い文を除いた記事も同様に学習させるものとする。また、本研究では、既存のデータ拡張手法であるEDA[4](文書分類に適応したもの)を自動要約システムに適応させて、提案手法と比較検討する。生成型要約は、Pointer-Generatorモデル[5]を用いた。データ拡張の方法(提案手法、EDA)については、2節で述べる。実験と評価は3節、結論は4節にて述べる。

2 データ拡張手法

2.1 提案手法

提案手法は、本来の学習に加えて入力記事にある操作をしてできた拡張記事と参照要約のペアを学習するというものである。入力記事に加える操作は以下の通りである。

1. 全データセットからトピックモデル[6]を作る
2. 入力記事の各文の重要度をトピックモデルから決める(文の重要度の決め方については後述する)
3. 最も重要度の低い文を入力記事から削除
4. 入力記事全てにおいて、2,3を繰り返す

文の重要度の決め方[7]は次のように行う。

1. あるトピックにおける文の重要度を計算する

(ア) あるトピックにおける文中の全ての単語の出現確率を求め、総和する

(イ) 文に使われた単語の数の平方根で割る

2. 全てのトピックで総和する

2.2 EDA

本研究では、データ拡張の比較対象として EDA を用いた。記事を直接変えて拡張する方法として、EDA を選んだ。EDA は以下の 4 つの手法からなる。

・ sr(synonym replacement)

同義語に置き換える

・ ri(random insertion)

同義語を記事のランダムな位置に挿入する

・ rs(random swap)

ランダムな 2 つの単語を入れ替える

・ rd(random deletion)

ランダムな単語を削除する

EDA では、上の 4 つの手法を行う頻度を記事に使われている単語数にハイパーパラメータ α を掛けて求めている。この研究では、訓練データが 5000 以上だと α は 0.1 に、データ拡張して増やす記事数を 4 倍としているので、それに倣った。そして、EDA を Pointer-Generator モデルに適応して、提案手法との比較検討を行った。

3. 実験と評価

3.1 データセット

本研究では、データセットとして、CNN/DailyMail dataset[8]を用いた。このデータセットは、訓練データ 287226 記事、評価データ 13768 記事、テストデータ 11490 記事が存在する。実験では、訓練データ数によって、データ拡張の効果が変わることを想定して、訓練データとして 10000 記事、45000 記事、90000 記事、180000 記事、287226 記事を用意した。

3.2 評価方法

自動要約の一般的な評価手法として、参照要約と生成した要約の単語の一致で評価する ROUGE[9]が用いられる。今回用いたのは、ROUGE-1、ROUGE-2、ROUGE-L である。ROUGE-1、ROUGE-2 はそれぞれユニグラム、バイグラムの一致で評価する。ROUGE-L は最長共通部分列の一致で評価する。そして、それぞれの ROUGE 評価手法には、F 値、recall、precision というものが存在する。recall とは、参照要約のうちにどれだけ生成した要約の単語が含まれているかで、precision というのは、生成した要約のうちにどれだけ参照要約の単語が含まれているかである。F 値は以下の式で求める。

$$F = \frac{2 \times Recall \times Precision}{Recall + Precision}$$

3.3 実験条件

本研究で使った Pointer-Generator プログラム[10]は Pytorch を使用したものとなっている。このプログラムでも、[5]と同等の結果を得られることは検証済みである。このプログラムでは、入力記事をエンコードする際に使用する単語数は、400 と限定されている。しかし、本研究で用いるデータ拡張の手法において、最初の単語から 400 単語までの内に、最も重要度の低い文が存在しない場合は、文を抜いたとしても、学習に何の影響もでない。そこで、学習データに使われている記事の中で、最も単語数の多い記事を見つけた。その単語数は 2380 語であった。そして、入力記事をエンコードする際に使用する単語数の上限を 2380 に設定した。表 1 に単語数の上限を 400 にした場合と 2380 にした場合の ROUGE の値の変動を示す。ただし、学習した記事数は、全記事データ、つまり 287,226 記事となっている。

表 1　単語数を 400 と 2380 にしたときの ROUGE の値

	ROUGE-1-f	ROUGE-1-r	ROUGE-1-p	ROUGE-2-f	ROUGE-2-r	ROUGE-2-p
400	0.3935	0.4372	0.3800	0.1709	0.1891	0.1662
2380	0.3958	0.4181	0.3994	0.1741	0.1832	0.1770

	ROUGE-L-f	ROUGE-L-r	ROUGE-L-p
400	0.3616	0.4014	0.3493
2380	0.3644	0.3846	0.3679

表 1 をみてもらうとわかるように、単語数の上限を 400 から 2380 にすることによる影響は表れていないことがわかる。

また、先行研究では、コピー機構で学習した後、カバレッジ機構で学習する。従って、本研究でもそれに倣う。本研究では、バッチサイズを 8 に設定した。表 1 では、バッチサイズの記事を何回学習するかを示している。

表 2　各記事数に対する学習回数

	iteration in copy mechanism	iteration in coverage mechanism
10,000 articles	20,000 times	300 times
45,000 articles	80,000 times	1,000 times
90,000 articles	160,000 times	2,000 times
180,000 articles	320,000 times	4,000 times
287,226 articles	460,000 times	6,000 times

この研究で使用される pointer-generator プログラムは、1 バッチ学習するのに約 0.27 秒かかった。本研究で使用した GPU は GeForce GTX 1080 Ti である。したがって、10,000 記事を学習するには約 1 時間半かかる。同様に、45000 記事、90000 記事、180000 記事、287226 記事を学習するのに、約 6 時間、12 時間、24 時間、35 時間かかる。拡張すると記事数が二倍になるため学習時間は二倍になるが、本研究では、記事数を二倍にするとき、バッチサイズを半分にしたためかかる時間は同じとなった。

3.4　実験結果

提案手法と EDA で拡張したときと、ベースの学習法の ROUGE による評価を表 3、表 4、表 5 に示す。ただし、表 3 は、すべて ROUGE-1 の F 値、表 4 は、すべて ROUGE-2 の F 値、表 5 は、すべて ROUGE-L の F 値となっている。また、normal はベースの学習法で、extended は提案手法となっている。

表 3、表 4、表 5 はいずれも、10,000 記事、45,000 記事、90,000 記事、180,000 記事で学習したものとなっている。

表 3　拡張手法とベースの学習法の比較

(ROUGE-1 の F 値)

	10,000articles	45,000articles	90,000articles	180,000articles
sr	0.3288	0.3359	0.3455	0.3703
ri	0.3235	0.3445	0.3381	0.3743
rs	0.3393	0.3456	0.3496	0.3752
rd	0.3305	0.3209	0.3528	0.3705
normal	0.3226	0.3398	0.3538	0.3758
extended	0.3335	0.3506	0.3627	0.3827

表 4　拡張手法とベースの学習法の比較

(ROUGE-2 の F 値)

	10,000articles	45,000articles	90,000articles	180,000articles
sr	0.1247	0.1294	0.1363	0.1566
ri	0.1186	0.1322	0.1306	0.1572
rs	0.1294	0.1338	0.1338	0.1576
rd	0.1243	0.1206	0.1414	0.1547
normal	0.1209	0.1304	0.1404	0.1587
extended	0.1275	0.1378	0.1495	0.1628

表 5　拡張手法とベースの学習法の比較

(ROUGE-L の F 値)

	10,000articles	45,000articles	90,000articles	180,000articles
sr	0.3044	0.3110	0.3212	0.3430
ri	0.2993	0.3194	0.3144	0.3456
rs	0.3145	0.3209	0.3248	0.3465
rd	0.3046	0.2982	0.3283	0.3437
normal	0.2994	0.3155	0.3288	0.3475
extended	0.3085	0.3249	0.3372	0.3515

287,226 記事(全データ)で学習したときの、提案手法とベースの学習法の ROUGE の値を表 6 に示す。

表 6　提案手法とベースの学習法の比較

(287,226 記事)

	ROUGE-1-f	ROUGE-1-r	ROUGE-1-p	ROUGE-2-f	ROUGE-2-r	ROUGE-2-p
normal	0.3918	0.4121	0.3958	0.1703	0.1784	0.1733
extended	0.3941	0.4288	0.3859	0.1721	0.1869	0.1693

	ROUGE-L-f	ROUGE-L-r	ROUGE-L-p
original	0.3593	0.3776	0.3633
extended	0.3625	0.3941	0.3551

3.5 考察

記事数が増え 90,000 記事、180,000 記事になると、EDA の拡張手法は、ベースの学習法よりも悪い結果となっている。一方で提案手法はどの記事数でもベースの学習法よりも良い結果となっている。ただ、10,000 記事のときは、EDA の rs 手法が提案手法を上回っている。また、拡張の効果は、記事数が少ない方のときの方が出ることがわかる。試しに、提案手法とベース

の学習法の、ROUGE の値の差を測ってみると、表 7 のようになる。(提案手法の ROUGE の値から、normal の ROUGE の値を引く)

表 7　提案手法(extended)からベースの学習法(normal)の ROUGE の値を引いた結果

	10,000articles	45,000articles	90,000articles	180,000articles	287,226articles
ROUGE-1-f	0.0109	0.0108	0.0089	0.0069	0.0023
ROUGE-2-f	0.0066	0.0074	0.0091	0.0041	0.0018
ROUGE-L-f	0.0091	0.0094	0.0084	0.0040	0.0032

このことからも、記事数が少ないときほど拡張の効果が出ることがわかる。

4　結論

本研究で、提案手法による拡張手法が有効であることが示せた。特に、少ない記事数のときに拡張の効果は出るが、そのときでも、EDA の手法よりもおおよそ良い結果となった。

今後の課題として言えるのは、まず他の自動要約システムにも、提案手法が適応できないかどうかというのと、どれくらい効果があるのかを調べることである。また、新たな拡張手法も考えられる。本研究では、どのような長さの記事に対しても一文を除くだけであったが、EDA のように、記事の長さによって変更できれば、もう少し改善がみられるのではないかと考えている。他にも、文の重要度の測り方として ROUGE を使う方法や、拡張手法として、重要な文だけを残す方法などが考えられる。

参考文献

[1] Data Growth,Business Opportunities, and the IT Imperatives, http://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm

[2] Dongxu Zhang, Zhichao Yang, “Word Embedding Perturbation for Sentence Classification” arXiv:1804.08166vl(2018)

[3] I. Saito, J. Suzuki, K.Nishida, K. Sadamitsu, S. Kobashikawa, Ryo. Masumura, Y. Matsumoto, J. Tomita “Improving Neural Text Normalization with Data Augmentation at Character- and Morphological Levels” Proceedings of the 8th International Joint Conference on Natural Language Processing,pages257-262(2017)

[4] Jason Wei, Kai Zou “EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks” arXiv:1901.11196v2[cs.CL](2019)

[5] Abigail. See, Peter J. Liu, et al. “Get To The Point: Summarization with Pointer-Generator Networks” arXiv:1704.04368(2017)

[6] Blei,D.M，Ng，A.Y. and Jordan，M.I. “Latent Dirichlet Allocation”，journal of Machine Learning Research，3，pp.993-1022，(2003)

[7] 重松　遥、　小林　一郎、“潜在的トピックの重要度を考慮した要約文の生成”言語処理学会　第 18 回年次大会　発表論文集(2012)

[8] https://cs.nyu.edu/~kcho/DMQA/

[9] Lin C. W., “ROUGE: A Package for Automatic Evaluation of Summaries”, Proceeding of the ACL-04 workshop, Vol.8 (2004)

[10]https://github.com/atulkum/pointer_summarizer