系列ラベリングタスクのための単純なデータ水増し

山崎智弘
東芝研究開発センター AI アナリティクスラボラトリー
tomohiro2.yamasaki@toshiba.co.jp

概要

語順が重要な系列ラベリング系のタスクに適用できる汎用的な水増しとして，cycle と randpad という新たな手法を提案する．どちらの手法も語順を維持したまま水増しする点に特徴がある．

2 つのデータセットを対象に，提案手法で水増ししたデータで系列ラベリングモデルを学習する実験を行なった．提案手法は大規模な言語モデルや外部知識がなくても適用できる非常にお手軽な水増しであるにも関わらず，ベースラインよりも 1 ポイントほど性能向上したことが確かめられた．

1 はじめに

日本の労働人口は少子化のため急速に減少しつつあり，製造業の現場ではベテランの持っていたノウハウの喪失が深刻な問題となっている．製造プラントは数多くの設備や機器から構成されており，運用・保守では長年の経験が重要なためである．

片や製造業では，日々の業務で起こったトラブルを報告書として記録しておく取り組みがある．そこで我々はベテランのノウハウを形式知化するため，自然言語処理 (NLP) を用いてトラブル報告書を構造化する技術開発を進めている．報告書にはどんなトラブルが起こったかだけでなく，どんな対処をしたか，その対処で解決したかなども記録されているため，構造化してトラブルを解決する可能性の高い対処をまとめられれば運用・保守のコスト低減につながるほか，若手の教育にも役立つと考えられる．

我々は報告書のテキストに含まれるトラブル表現を教示した訓練データを用意し，ニューラルネットワークによる深層学習で固有表現抽出 (NER) を行なう方針を取っている．深層学習はコンピュータビジョンの分野で発展してきたが，近年は NLP の分野でも文書分類，機械翻訳，質問応答などさまざまなタスクで最高性能を達成し続けているためである．

深層学習の性能は一般に，訓練データの質と量によるところが大きいと言われている．しかしトラブル報告書は専門知識なしでは理解しづらいうえ，どこからどこまでをトラブル表現とみなすかは作業者ごとに揺れが生じやすいため，良質で大量の訓練データを用意することが難しい．

画像や音声であれば，例えば画像を回転・反転しても映っている物体は変化しないという知識に基づいて機械的に訓練データを水増しできるが，NLP ではどのようなドメインの文にも意味を保ったまま機械的に適用できる変換がないため，水増しのアプローチは限定的である．

従来よく用いられてきたのは同義語による置換だが，意味を保つためのヒューリスティクスに頼りがちである．EDA [1] は単語のランダムな編集を組み合わせることで単純ながら汎用的な水増しをある程度実現しているが，語順が重要な系列ラベリング系のタスクには適用が難しい．

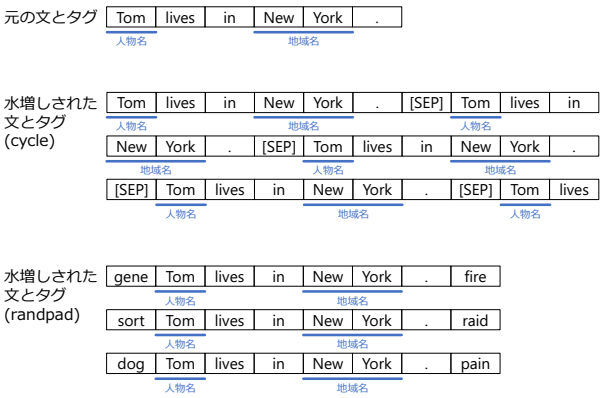

図 1 提案手法による訓練データの水増し

そこで我々は，語順が重要な系列ラベリング系のタスクにも適用できる汎用的な水増しとして，cycle と randpad という新たな手法を提案する．cycle は開始位置をシフトしながら各文を何度も繰り返すことで，randpad は各文の先頭と末尾に適当な単語をランダムに付加することでデータを水増しする．水増しは学習フェーズの訓練データにだけでなく推定フェーズの評価データにも適用し，水増しされたそ

れぞれのデータに対する推定結果から元のデータの推定結果を得る．

以下では cycle と randpad のアルゴリズムを説明し，一般的な NER の公開データセット (英語) と我々の持っている電力プラントのトラブル報告書 (日本語) で行なった実験について述べる．これらの実験の評価結果から，提案手法は非常にお手軽な水増しであるにも関わらず，ベースラインよりも性能向上することを示す．

2 関連研究

機械翻訳タスクでは，別言語に翻訳して元言語に逆翻訳する手法 [2] が水増しとして有効である．しかしトラブル表現は専門用語が多く，うまく機械翻訳できないことも多い．

同義語の置換という観点からは，WordNet [3] のような人手で作られたオントロジーを用いる手法，埋め込みベクトルが近いものを同義語の候補とする手法 [4] などがある．NER では同じクラスの固有表現を同義語の候補とすることも行なわれるが，それだけでは固有表現を含まない文は何も置換できないので他の手法と併用する必要がある．ドメイン固有の専門用語に対しては，オントロジーにないため候補が得られない，あるいは単語の使われる文脈が異なるので埋め込みベクトルが近くても意味が異なる，などの課題があり，いずれの手法も候補の選び方が恣意的になりやすい．

一方 BERT [5] で同義語の候補を選ぶ CDA [6] という手法がある．BERT は文脈を考慮したよい埋め込みベクトルを得る手法として知られているが，CDA は BERT が文脈を考慮して予測した単語による置換に制限することで文の意味を保つようにしている．

単語レベルではなく特徴ベクトルのレベルで訓練データを水増しする手法 [7, 8] も提案されている．これらは，訓練データから抽出された 2 組の入出力の線形補完によって水増しする Mixup [9] という手法を NLP に持ち込んだものである．ただしこれらの手法も，文の意味を保つように入力文の意味が似ている組に抽出を制限している．

反対に，文の意味が多少変わることを許容することで，非常にお手軽な水増しを実現している EDA [1] という手法がある．同義語による置換のほか，単語のランダムな追加・削除・交換を組み合わせて水増しする．大規模な言語モデルや外部知識がなくても適用できるので，訓練データが少ないときの文書分類や極性判定の性能をほとんどコストをかけずに底上げできることが知られている．

3 提案手法

本節では語順を維持したまま水増しする cycle と randpad の具体的なアルゴリズムを説明する．

入力は文の集合 $\{\mathrm{sent}_i \mid 0 \le i < I\}$ であり，それぞれの文 sent_i は J_i 単語からなる系列 $w_{i,0}, \ldots w_{i,J_i-1}$ で表され，固有表現は単語列に対してタグ付けされた範囲で表されるものとする．

例えば図 1 上側は，6 単語で表される文の $[0,1)$ および $[3,5)$ の範囲にそれぞれ人物名と地域名のタグが付いていることを表す．実際にはタグに B や I をつけて範囲の先頭とそれ以外を区別することも多いが，以下ではその区別は捨象して説明する．

3.1 cycle

図 1 中側は cycle による水増しの概念図である．開始位置をシフトしながら各文を何度も繰り返す．

まずそれぞれの文 sent_i ごとに，文を繰り返す領域の大きさとして単語数 J_i より大きい適当な $f(J_i)$ を設定する．J_i によらない値 (例えば $\max_i J_i + 1$) に設定するとパディングが不要になるので学習が簡単になるが，長い文に比べて短い文の繰り返しが多くなり，モデルが偏る可能性がある．そこで適当な係数 a, b を用いて $f(J_i) = aJ_i + b$ などとする．

次に水増し倍率 n_{aug} が与えられたとしよう．文を繰り返すときの開始位置はランダムに与えてもよいが，均等に $c_i^k = \lfloor k f(J_i)/n_{\mathrm{aug}} \rfloor$ (ただし $0 \le k < n_{\mathrm{aug}}$) などとする．$c_i^k$ が同じ値を取らないように $f(J_i) \ge n_{\mathrm{aug}}$ にしておくことが望ましい．

c_i^k が定まれば，そこから順に $f(J_i)$ の領域を文尾と文頭がつながらないようにセパレータ [SEP] を挟みながら文 sent_i の単語で繰り返し埋めていくことで，水増しされた文 sent_i^k が得られる．すなわち文 sent_i^k を構成する単語 $w_{i,j}^k$(ただし $0 \le j < f(J_i)$) は具体的には $w_{i,j-c_i^k \pmod{J_i+1}}$ と表される．

学習フェーズでは，元の文のタグ範囲に対応する水増しされた文の範囲にタグ付けして水増しされた訓練データとする．水増しされた文の単語と元の文の単語の対応は取れているので，元の文の単語のタグをそのまま付ければよい．

推定フェーズでは，水増しされた文に対して推定処理を行ない，それぞれの単語のタグを推定する．水増しされた文の単語と元の文の単語の対応が取れ

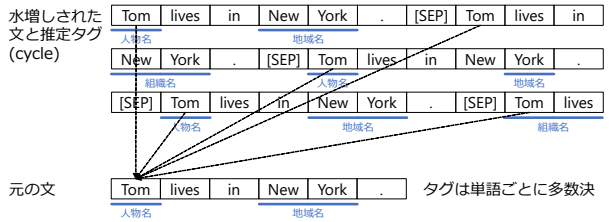

図 2　cycle の推定フェーズ

ているので，元の文の単語に付くタグは水増しされた文の単語に付いたタグによる投票で決定できる．

図 2 の例であれば，水増しされた文において “Tom” は 3/5 が人物名なので人物名，“New” は 2/4 が地域名なので地域名となる．この例では “York” は地域名も組織名も 2/4 となるので多数決ではどちらを選んでもよいが，つながりがよいように “New York” をまとめて地域名とする．

3.2　randpad

図 1 下側は randpad による水増しの概念図である．各文の先頭と末尾に適当な単語をランダムに付加する．文頭と文尾に付加する単語は文の意味を保つようにする必要があるため，すべての訓練データで全くタグが付いていない単語の集合 $W_{\mathrm{O}} = \{w \mid w \text{ のタグが O}\}$ を事前に求めておく．

まず文頭と文尾に付加する単語数 n_{pad} を設定する．付加する単語数は文頭と文尾で異なっていてもよいが，双方向言語モデルを用いるので対称にするため同じ値にするものとする．

次に水増し倍率 n_{aug} が与えられたとしよう．各文の先頭と末尾に W_{O} から n_{pad} 単語ずつをランダムに選んで付加すると水増しされた文 sent_i^k が得られるので，それを n_{aug} 回繰り返せばよい．すなわち文 sent_i^k を構成する単語 $w_{i,j}^k$(ただし $0 \le j < J_i + 2n_{\mathrm{pad}}$) は，具体的には $n_{\mathrm{pad}} \le j < J_i + n_{\mathrm{pad}}$ のとき $w_{i,j-n_{\mathrm{pad}}}$ でそれ以外のときランダムな単語となる．

学習フェーズでは，元の文のタグ範囲に対応する水増しされた文の範囲にタグ付けして水増しされた訓練データとする．水増しされた文の単語と元の文の単語の対応は取れているので，元の文の単語のタグをそのまま付ければよい．付加した単語は W_{O} から選ぶのでタグを付けないものとする．

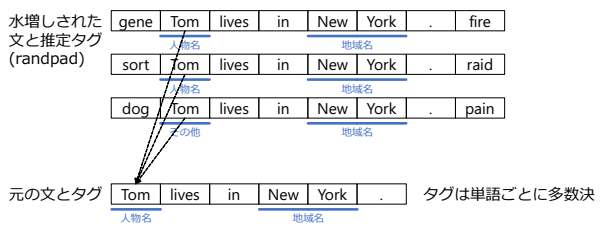

図 3　randpad の推定フェーズ

推定フェーズでは，水増しされた文に対して推定処理を行ない，それぞれの単語のタグを推定する．文頭と文尾から n_{aug} 単語を取り除けば元の文の単語になるので，元の文の単語に付くタグは水増しされた文の単語に付いたタグによる投票で決定できる．

図 3 の例であれば，水増しされた文において “Tom” は 2/3 が人物名なので人物名となり，“New” と “York” は 3/3 が地域名なので地域名となる．

4　実験と評価

今回の実験では系列ラベリングモデルとして BiLSTM-CRF [10] を用いた．具体的なネットワークは，単語の埋め込みベクトルを求める Embedding 層，確率 0.5 の Dropout 層，入出力の次元数が等しい Linear 層，256 次元の BiLSTM 層，タグに対応する出力の次元数を持つ CRF 層から構成される．なお 768 次元の事前学習済み BERT [1)] の末尾 4 層を concat して用いたので，Embedding 層の次元数は $3,072 = 768 \times 4$ である．

学習に用いたバッチサイズは 256，損失関数は CRFLoss，オプティマイザは SGD である．学習率は 10^{-1} から検証データの損失が 4 エポック下がらないたび半減し，10^{-5} を下回った時点で早期終了する．

以下では訓練データの水増しを行なわないとき (baseline) と，文を繰り返す領域の大きさ $f(J_i)$ を $64\lceil J_i/48 \rceil$ とした cycle，付加する単語数 n_{pad} を 1, 2 とした randpad をそれぞれ適用したときの性能を比較した．なお baseline 以外は，水増し倍率 n_{aug} を 1, 5, 10 としたときの性能も検証した．

表 1　CoNLL03 のデータ数の内訳

	文	人物名	地域名	組織名	その他
訓練	14,986	6,600	7,140	6,321	3,438
検証	3,465	1,842	1,837	1,341	922
評価	3,683	1,617	1,668	1,661	702
平均長	13.68	1.69	1.16	1.57	1.34

はじめに，NER の公開データセットである CoNLL03 [11] を用いて提案手法による水増しの効果を評価した結果について示す．本データセットは人物名・地域名・組織名・その他の 4 種のタグが付いており，データ数は表 1 のとおりである．平均長は文や固有表現を構成する単語数の平均を表す．

1)　bert-base-uncased(英)，cl-tohoku/bert-base-japanese-v2(日)

表 2 は系列ラベリングモデルによって抽出されたタグ範囲が正解と完全一致するかどうかで評価したときの F 値である．それぞれの設定で 3 回ずつ学習したモデルによる平均と標準偏差を示す．

表 2 からわかるように，randpad2 は性能向上がほとんど見られないが，cycle と randpad1 は $n_{\text{aug}} = 5, 10$ のとき baseline を上回る．いずれも n_{aug} を増やすほど上がり幅も大きくなっているので，cycle は $f(J_i)$ の選び方によっても変わると思われるが，さらに増やせばより性能向上する可能性がある．

表 2 CoNLL03 を用いたときの評価結果

	$n_{\text{aug}} = 1$	5	10
baseline	.908 ± .002	—	—
cycle	.908 ± .001	.912 ± .001	.912 ± .003
randpad1	.906 ± .001	.912 ± .001	**.913 ± .002**
randpad2	.901 ± .001	.909 ± .003	.908 ± .002

一方，baseline で抽出されたが randpad で抽出されなかった事例を分析すると，W_{O} に含まれる単語を含むものが多いことがわかった．W_{O} から選んで付加した単語はタグが付かないように学習するので，その影響を受けた可能性がある．randpad1 で性能が上がったのに randpad2 で上がらなかったのは，下げる効果の方が大きかったためであろう．

逆に言えば，付加する単語の選び方を工夫したり，WordDropout [12] のように Embedding 層で埋め込みベクトルをランダムに生成したりすれば，randpad はより性能向上する可能性がある．

続いて，我々の持っている電力プラントのトラブル報告書を用いて提案手法による水増しの効果を評価した結果について示す．トラブル表現 (何がどうなった，何をどうした) にイベントというタグが付いており，データ数は表 3 のとおりである．平均長は文やイベントを構成する単語数の平均を表す．

表 3 トラブル報告書のデータ数の内訳

	文	イベント
訓練	7,145	8,001
検証	794	919
評価	555	457
平均長	31.49	13.36

CoNLL03 での実験のように，NER の評価は完全一致で行なうのが一般的である．しかしイベントのタグ範囲は非常に長く，また前述のとおり正解に揺れが生じやすいため，完全一致にそこまでこだわる必要はない．イベントの主要部 (どうなった，どうした) は末尾付近に出現することが多いので，以下の評価では [13] と同じく，タグ範囲の末尾 5 単語に重なりがあるかどうか (主要部一致) で行なうものとした．例えば「建屋の配管に亀裂が発生。」という文の正解が「配管に亀裂」だった場合，「建屋の配管に亀裂」や「亀裂が発生」は少なくとも「亀裂」という重なりがあるので OK とみなす．

表 4 は系列ラベリングモデルによって抽出されたタグ範囲が正解と主要部一致するかどうかで評価したときの F 値である．それぞれの設定で 3 回ずつ学習したモデルによる平均と標準偏差を示す．

表 4 トラブル報告書を用いたときの評価結果

	$n_{\text{aug}} = 1$	5	10
baseline	.832 ± .002	—	—
cycle	**.844 ± .004**	.835 ± .007	.833 ± .005
randpad1	.834 ± .004	.830 ± .006	.828 ± .002
randpad2	.832 ± .003	.839 ± .003	.827 ± .003

表 4 からわかるように，cycle は baseline を上回るものの randpad1 と randpad2 は性能向上がほとんど見られない．付加した単語はタグが付かないように学習したため，CoNLL03 での実験と同じく性能を下げる効果の影響を受けた可能性がある．下がり幅が抑えられているのは，タグが 1 種しかないのでタグを間違えるエラーがないことや主要部一致で評価したことが要因であろう．

5 おわりに

本論文では，語順が重要な系列ラベリング系のタスクにも適用できる汎用的な水増しとして，cycle と randpad という新たな手法を提案した．cycle は開始位置をシフトしながら各文を何度も繰り返すことで，randpad は各文の先頭と末尾に適当な単語をランダムに付加することで語順を維持したままデータを水増しする手法であり，大規模な言語モデルや外部知識がなくても適用できる．

CoNLL03 およびトラブル報告書のデータセットを提案手法で水増しして BiLSTM-CRF を学習する実験を行なったところ，ベースラインよりも 1 ポイントほど性能向上したことが確かめられた．ただし randpad はパラメタによってはほぼ変わらないか逆に下がることも確かめられた．

今後はネットワーク構造や与えられたデータごとに最適な水増しパラメタを求め，理論解析を通じてさらによい水増し手法を開発していく予定である．

参考文献

[1] Jason Wei and Kai Zou. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. 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)**, pp. 6382–6388, Hong Kong, China, November 2019. Association for Computational Linguistics.

[2] Marzieh Fadaee, Arianna Bisazza, and Christof Monz. Data augmentation for low-resource neural machine translation. In **Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)**, pp. 567–573, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[3] George A. Miller. Wordnet: A lexical database for english. **Communications of the ACM**, Vol. 38, No. 11, pp. 39–41, 1995.

[4] William Yang Wang and Diyi Yang. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets. In **Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing**, pp. 2557–2563, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)**, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[6] Sosuke Kobayashi. Contextual augmentation: Data augmentation by words with paradigmatic relations. In **Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)**, pp. 452–457, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[7] Rongzhi Zhang, Yue Yu, and Chao Zhang. SeqMix: Augmenting active sequence labeling via sequence mixup. In **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**, pp. 8566–8579, Online, November 2020. Association for Computational Linguistics.

[8] Jiaao Chen, Zhenghui Wang, Ran Tian, Zichao Yang, and Diyi Yang. Local additivity based data augmentation for semi-supervised NER. In **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**, pp. 1241–1251, Online, November 2020. Association for Computational Linguistics.

[9] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In **6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings**. OpenReview.net, 2018.

[10] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In **Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**, pp. 260–270, San Diego, California, June 2016. Association for Computational Linguistics.

[11] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In **Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003**, pp. 142–147, 2003.

[12] Rico Sennrich, Barry Haddow, and Alexandra Birch. Edinburgh neural machine translation systems for WMT 16. In **Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers**, pp. 371–376, Berlin, Germany, August 2016. Association for Computational Linguistics.

[13] 伊藤雅弘, 山崎智弘. アノテーション漏れ推定を用いたエンティティ抽出. 言語処理学会 第 27 回年次大会 発表論文集, pp. 1264–1268, 3 2021.