HTML 構造を補助情報として利用する 日本語ブログ記事からの固有表現抽出

植 畳 [1]　數見 拓朗 [2]　小泉 和之 [1]
[1] 横浜市立大学大学院　[2] 株式会社サイバーエージェント
[1]{w205607d,zumi}@yokohama-cu.ac.jp　[2]kazumi_takuro@cyberagent.co.jp

概要

Web 上のテキストからの固有表現抽出では，HTML タグが除去され自然言語のテキストのみが利用されていることが多い．本研究では，HTML 構造を補助情報として利用する固有表現抽出手法を提案する．提案手法は複数あり，グラフニューラルネットワークを用いて HTML 構造を読み取るアプローチの手法と，単語と HTML タグの系列を既存の固有表現抽出モデルに入力するアプローチの手法に分けられる．また，ブログ記事を用いた実験を通して，提案手法の中でも BERT と Graph Attention Networks を組み合わせた手法と BERT に HTML タグを入力する手法の 2 つが優れた性能を持つことを示す．

1　はじめに

現在，Web 上には様々なテキストが蓄積されており，その 1 つとしてブログ記事が挙げられる．2000 年以降のブログサービスの急速な普及に伴い，日々増加する大量の記事から有用な情報を抽出する技術が求められ，盛んに研究されている [1, 2, 3, 4].

本研究では，HTML テキストとして記述された日本語ブログ記事からの固有表現抽出を扱う．HTML テキストからの固有表現抽出に関する先行研究では，前処理として HTML タグを除去し自然言語のテキストのみを利用していることが多い [5, 6]．そこで本研究では，HTML 構造を補助情報として利用する固有表現抽出手法を提案しその有効性を検証する．

本研究では複数の手法を提案するが，それらは大きく 2 種類に分けられる．1 つ目は，既存の固有表現抽出モデルに HTML タグをトークンとして入力する単純なアプローチである．2 つ目は，グラフニューラルネットワークを用いて HTML 構造情報を処理するモジュールと既存の固有表現抽出モデルを用いて自然言語情報を処理するモジュールとを組み合わせるアプローチである．

レシピに関するブログ記事を対象とした実験を行い，提案手法を用いて HTML 構造を補助情報として活用することで既存手法に比べて性能が向上することを確認した．提案手法の中でも特に，BERT [7] と Graph Attention Networks [8] とを組み合わせた手法と，BERT に HTML タグをトークンとして入力する手法の 2 つが高い性能を示すことを確認した．

2　関連研究

近年，ニューラルネットワークを用いた固有表現抽出モデルが注目を集め盛んに研究されている．Lample *et al.* [9] は双方向の LSTM と Conditional Random Field (CRF) を組み合わせた BiLSTM-CRF モデルを提案し，この手法は現在の固有表現抽出の主流となり様々な自然言語データに適用されている．また，自然言語処理分野において汎用的なモデルである BERT [7] をベースにした手法が，いくつかのデータセットにおいて最高性能を記録する [10] などして注目を集めている．Web 上のテキストを扱った研究の例として，Duc *et al.* [6] が提案した Web 上から固有表現を収集するシステムでは，Google 検索によって収集した Web ページに対して HTML タグを削除した上で固有表現抽出モデルが適用されている．

3　手法

3.1　HTML タグをトークンとして入力するアプローチ

提案手法 1：BiLSTM-CRF with Tag　この提案手法では，HTML テキストを「単語と HTML タグが入り混じった系列」と見なし，それを埋め込みベクトル系列に変換し BiLSTM-CRF [9] に入力する．例えば

「今日はハンバーグを作る」という文があった場合，先行研究では HTML タグを除去した系列「今日/は/ハンバーグ/を/作る」を BiLSTM-CRF に入力する場合が多い（例えば [5] など）が，この提案手法では「今日/は//ハンバーグ//を/作る」というそのままの系列を入力する．各トークンの埋め込みベクトルは，単語については東北大学の研究グループによる日本語 Wikipedia エンティティベクトル [11] で初期化した上でファインチューニングを行い，各種 HTML タグについては乱数で初期化した上で学習させる．なお，HTML タグ内の属性は全て事前に除去し利用せず，開始タグと終了タグは区別して扱う．

提案手法 2：BERT with Tag　この提案手法では，前項の手法と同様に HTML テキストを「単語と HTML タグが入り混じった系列」と見なし，そのまま BERT [7] に入力する．単語の埋め込みベクトルと BERT 内部の Transformer Encoder [12] のパラメータについては東北大学の研究グループによる事前学習済み BERT モデル [13] で初期化した上でファインチューニングを行い，各種 HTML タグの埋め込みベクトルは乱数で初期化した上で学習させる．

3.2 グラフニューラルネットワークを用いて HTML 構造を読み取るアプローチ

本節では，グラフニューラルネットワーク (GNN) を用いて HTML 構造情報を処理するモジュールと既存の固有表現モデルを用いて自然言語情報を処理するモジュールとを組み合わせるアプローチの提案手法を 3 つ説明する．

HTML テキストのグラフ表現　Document Object Model (DOM) [14] は，マークアップされたテキストをツリー構造で表現し操作する仕組みである．HTML テキストは HTML タグでマークアップされたテキストであるため，DOM を用いることでツリー構造のグラフとして表現することができ，これを本論文では DOM グラフと呼ぶこととする．DOM グラフのノードは要素ノードとテキストノードの 2 つに大別できる．例えば「<h1>見出し </h1>」という HTML テキストは，要素ノード「h1」とテキストノード「見出し」がエッジで繋がった DOM グラフとして表される．本研究で扱うブログ記事の HTML テキストとそれを DOM グラフとして表したものを図 1 の左側に示す．

GNN によるノード埋め込み　抽出したい固有表現はグラフ中の一部のテキストノードに含まれる．そこで本節の提案手法では，GNN を利用し HTML 構造情報を考慮した上で各テキストノードをベクトルに埋め込むことを考える．まず，ノード埋め込みを行う様々な GNN の共通事項を述べる．ノード集合 V とエッジ集合 E の組からなるグラフ G において，各ノードの特徴量ベクトル $\boldsymbol{x}_v[v \in V]$ を用意する．最終的な出力は各ノードの埋め込みベクトル $\boldsymbol{y}_v^{\mathrm{GNN}}[v \in V]$ である．また，内部に学習可能なパラメータ Θ を持ちグラフ構造を考慮した変換を行う関数 $f(\cdot; G, \Theta)$ をレイヤ数の分だけ複数用意する．この関数 f はグラフ上の近傍ノードの特徴量ベクトルを集約するように構成され，具体形は手法によって異なる．また，σ を活性化関数とし，ネットワークのレイヤ数は L とする．そして，次式のように更新していくことで，ノードの特徴量 $\boldsymbol{X}_V := \left[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{|V|}\right]^T$ を埋め込み表現 $\boldsymbol{Y}^{\mathrm{GNN}} := \left[\boldsymbol{y}_1^{\mathrm{GNN}}, \boldsymbol{y}_2^{\mathrm{GNN}}, \ldots, \boldsymbol{y}_{|V|}^{\mathrm{GNN}}\right]^T$ に変換する．

$$\boldsymbol{X}^{(0)} = \boldsymbol{X}_V \tag{1}$$

$$\boldsymbol{X}^{(l)} = \sigma\left(f\left(\boldsymbol{X}^{(l-1)}; G, \Theta^{(l)}\right)\right)\ [l = 1, 2, \ldots, L-1] \tag{2}$$

$$\boldsymbol{Y}^{\mathrm{GNN}} = \boldsymbol{X}^{(L)} = f\left(\boldsymbol{X}^{(L-1)}; G, \Theta^{(L)}\right) \tag{3}$$

次に，本節の 3 つの提案手法それぞれで利用する Graph Convolutional Networks (GCN) [15], GraphSAGE [16], Graph Attention Networks (GAT) [8] について説明する．GCN では，まずグラフ G の全ノードに自己ループ (自分自身と接続するエッジ) を追加する．その状態での隣接行列を $\widehat{\boldsymbol{A}}$, 次数行列を $\widehat{\boldsymbol{D}}$ とする．GCN では，式 (2), (3) の関数 f として，次式の f_{GCN} を用いてノード特徴量を変換する．

$$f_{\mathrm{GCN}}(\boldsymbol{X}) = \widehat{\boldsymbol{D}}^{-\frac{1}{2}}\widehat{\boldsymbol{A}}\widehat{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{X}\boldsymbol{W} \tag{4}$$

ここで重み行列 $\boldsymbol{W}$ は学習対象のパラメータである．GraphSAGE では複数のモデルが提案されており，その中で本研究では GraphSAGE-mean (GSAGE) を利用する．このモデルでは，式 (2), (3) の関数 f として，次式の f_{GSAGE} を用いて各ノードの特徴量ベクトル $v \in V$ を変換する．

$$f_{\mathrm{GSAGE}}(\boldsymbol{x}_v) = \boldsymbol{W}_1\boldsymbol{x}_v + \boldsymbol{W}_2\ \mathrm{MEAN}(\{\boldsymbol{x}_u | u \in \mathcal{N}'(v)\}) \tag{5}$$

ここで重み行列 $\boldsymbol{W}_1, \boldsymbol{W}_2$ は学習対象のパラメータであり，MEAN は平均ベクトルをとる操作を表す．また，$\mathcal{N}'(v)$ はノード v の近傍ノードをランダムサンプリングして得られる集合であり，これは計算コス

トの削減を目的としている．GAT では，式 (2) の関数 f として，次式の f_{GAT} を用いて各ノードの特徴量ベクトル $v \in V$ を変換する．

$$f_{\mathrm{GAT}}(\boldsymbol{x}_v) = \mathrm{CONCAT}\left(\boldsymbol{h}_v^{(1)}, \boldsymbol{h}_v^{(2)}, \ldots, \boldsymbol{h}_v^{(K)}\right) \tag{6}$$

$$\boldsymbol{h}_v^{(k)} = \sum_{u \in \mathcal{N}(v)} \alpha_{vu}^{(k)} \boldsymbol{W}^{(k)} \boldsymbol{x}_u \tag{7}$$

$$\alpha_{vu}^{(k)} = \frac{\exp e_{vu}^{(k)}}{\sum_{u' \in \mathcal{N}(v)} \exp e_{vu'}^{(k)}} \tag{8}$$

$$e_{vu}^{(k)} = \mathrm{LReLU}\left(\boldsymbol{a}^{(k)^T} \mathrm{CONCAT}\left(\boldsymbol{W}^{(k)} \boldsymbol{x}_v, \boldsymbol{W}^{(k)} \boldsymbol{x}_u\right)\right) \tag{9}$$

ここで重み行列 $\boldsymbol{W}^{(1)}, \boldsymbol{W}^{(2)}, \ldots, \boldsymbol{W}^{(K)}$ と縦ベクトル $\boldsymbol{a}^{(1)}, \boldsymbol{a}^{(2)}, \ldots, \boldsymbol{a}^{(K)}$ は学習対象のパラメータであり，K は Multi-head Attention のヘッド数である．また，$\mathcal{N}(v)$ はノード v の近傍ノードの集合であり，LReLU は活性化関数 Leaky ReLU [17] である．

提案手法 3, 4, 5：BERT + GCN, GSAGE, GAT　提案手法 3, 4, 5 は，言語情報を処理する BERT 部と HTML 構造情報を処理する GNN 部で構成される．その概要図を図 1 に示す．BERT 部では，HTML タグを除去して自然言語のみとなったテキストのサブワード系列 $w_1, w_2, \ldots, w_N$ を BERT に入力し，さらに出力次元が固有表現ラベル数となるように線形変換を行い，ベクトル系列 $\boldsymbol{y}_1^{\mathrm{BERT}}, \boldsymbol{y}_2^{\mathrm{BERT}}, \ldots, \boldsymbol{y}_N^{\mathrm{BERT}}$ を得る．これを行列 $\boldsymbol{Y}^{\mathrm{BERT}} := \left[\boldsymbol{y}_1^{\mathrm{BERT}}, \boldsymbol{y}_2^{\mathrm{BERT}}, \ldots, \boldsymbol{y}_N^{\mathrm{BERT}}\right]^T$ にまとめる．GNN 部では，まず HTML テキストを DOM グラフで表現し，ノード特徴量 $\boldsymbol{x}_v[v \in V]$ としてノード種別（h1 要素ノード, p 要素ノード, ..., テキストノード）の One-Hot ベクトルを用意する．このグラフを式 (1), (2), (3) からなる GNN に入力し，ノード埋め込み表現 $\boldsymbol{Y}^{\mathrm{GNN}} := \left[\boldsymbol{y}_1^{\mathrm{GNN}}, \boldsymbol{y}_2^{\mathrm{GNN}}, \ldots, \boldsymbol{y}_{|V|}^{\mathrm{GNN}}\right]^T$ を得る．ここで，埋め込み次元は固有表現ラベル数と一致するようにしておく．BERT 部の出力 $\boldsymbol{Y}^{\mathrm{BERT}}$ と GNN 部の出力 $\boldsymbol{Y}^{\mathrm{GNN}}$ を合わせるために，サブワードと DOM グラフ上のノードの対応を表す行列 $\boldsymbol{I}_{\mathrm{Word,Node}}$ を用意する．この行列の第 i, j 成分は「第 i サブワードが第 j ノードに含まれていれば 1 でそうでなければ 0」となっている．そして，次式を本提案手法の最終的な出力とする．

$$\boldsymbol{Y}^{\mathrm{BERT+GNN}} = \boldsymbol{Y}^{\mathrm{BERT}} + \boldsymbol{I}_{\mathrm{Word,Node}} \boldsymbol{Y}^{\mathrm{GNN}} \tag{10}$$

$\boldsymbol{Y}^{\mathrm{BERT}}$ と $\boldsymbol{Y}^{\mathrm{GNN}}$ とをこのように組み合わせることで，各サブワードに対して BERT が出力したスコアベクトルに，そのサブワードが属するノードの埋め込み表現ベクトルが足し合わされ，言語情報と HTML 構造情報とを両方とも利用した固有表現抽出が行われる．ここで，上記の GNN 部に GCN, GSAGE, GAT を用いた 3 パターンの提案手法をそれぞれ BERT + GCN, BERT + GSAGE, BERT + GAT と呼ぶ．

表 1　各モデルの F1 スコア

モデル	全記事	ブログ A のみ	ブログ B のみ
BiLSTM (既存手法)	70.1	50.4	74.6
BiLSTM with Tag	75.3	54.4	77.6
BERT (既存手法)	78.5	72.5	80.2
BERT with Tag	80.2	72.6	**86.9**
BERT + GCN	80.3	76.4	80.8
BERT + GSAGE	80.9	76.6	80.4
BERT + GAT	**81.2**	**78.0**	81.6

4　実験と考察

4.1　データセット

本節では，提案手法の性能評価実験のために構築したデータセットについて述べる．まず，ブログサービス「アメーバブログ[1)]」上の料理レシピに関する 2 つのブログ A, B [2)] から，それぞれ 100 個ずつの記事を HTML テキストとして取得した．次に，株式会社サイバーエージェントから提供を受けたアノテーションシステム Orion Annotator [18] を利用し，その 200 記事に 3 種類の固有表現「料理」「材料」「分量」をアノテーションした．

4.2　実験設定

前項で構築したデータセットを比率 6 : 2 : 2 で Train, Valid, Test に分割し，Train で訓練を行い Valid でハイパーパラメータをチューニングすることで，提案手法 1, 2, 3, 4, 5 それぞれのモデルを作成した．また，比較のために，HTML タグを除去したテキストを利用する BiLSTM-CRF, BERT のモデル（すなわち提案手法 1, 2 から HTML タグを削除したモデル）も作成した．そして，この 7 つのモデルを Test セットに適用し，F1 スコアで性能を評価・比較した．なお，同様の実験をブログ A, B それぞれの 100 記事のみを含むデータセットに対しても行った．

4.3　実験結果と考察

実験結果を表 1 に示す．表中の評価値において，BiLSTM with Tag が BiLSTM を上回っていることと，

1）　https://ameblo.jp/
2）　同じブロガーが投稿した記事の集合をブログと呼ぶ

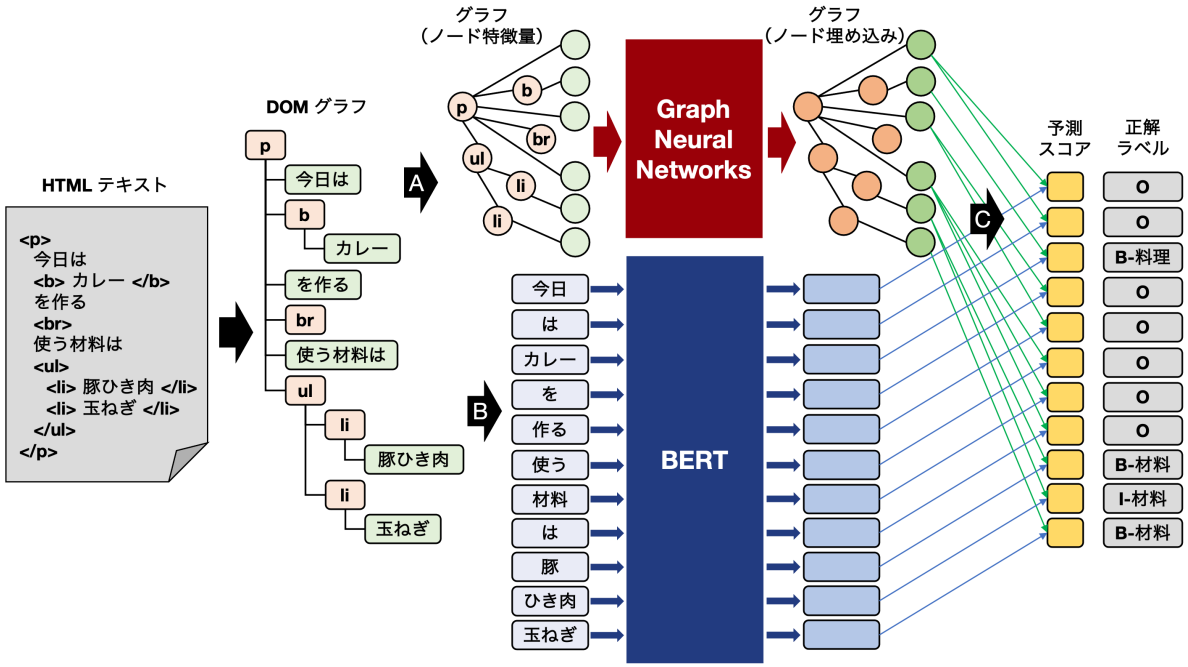

図 1　提案手法 3, 4, 5 の概要図．GNN 部に GCN, GSAGE, GAT レイヤを用いた手法をそれぞれ BERT + GCN, BERT + GSAGE, BERT + GAT と呼ぶ．図中の A ではノード特徴量としてノード種別の One-Hot ベクトルを持つグラフを作成している．B では各テキストノードに含まれる文を BERT のトークナイザでサブワード化している．C では各サブワードの BERT 出力ベクトルに対してそのサブワードが属していたノードの GNN 埋め込みベクトルが足し合わされている．

BERT with Tag, BERT + GCN, BERT + GSAGE, BERT + GAT が BERT を上回っていることから，提案手法を用いて HTML 構造を補助情報として活用することで既存手法に比べて性能が向上することを確認した．また，提案手法の中でも特に，BERT + GAT と，BERT with Tag の 2 つが高い性能を示している．

BERT + GCN, BERT + GSAGE, BERT + GAT はいずれも GNN を用いて HTML 構造情報を読み取るが，表 1 で性能を比較すると BERT + GAT が他 2 つを上回っている．これは，GAT が持つ Multi-head Attention 機構によるものだと考えられる．HTML テキストの DOM グラフには，異質なノードが混在している．例えば，太字を表す b ノード, 画像の存在を表す img ノード, 段落を表す p ノード, ハイパーリンクを表す a ノード, テキストノードなどである．このようなノードの異質性に対して Attention 機構の「異なる近傍ノードに異なる重みを与えて集約する」という仕組みが有効に作用したと考えられる．

また，ブログ B のみを対象とした実験では，ブログ A とは異なり BERT with Tag の性能が BERT + GAT を上回っている．ここで，固有表現のうち HTML の開始タグ（<b> や <p> など）の直後に現れているもの比率を調べたところ，ブログ A では 20.4 %, ブログ B では 52.0 % であったため，ブログ B では固有表現が開始タグの直後に現れやすい傾向があると言える．BERT + GAT では HTML 情報を DOM として扱うため，開始タグとその中の単語との位置関係を参照することはできず，先程のような傾向を利用することはできない．一方 BERT with Tag では HTML タグと単語の系列を BERT に入力するため，先程のような傾向を学習し利用することができる．このことが，ブログ B において BERT with Tag が BERT + GAT を上回った理由の 1 つとして考えられる．

5　おわりに

本研究では，Web 上のテキストを対象とした固有表現抽出タスクにおいて，HTML 構造を補助情報として利用する手法を提案した．そして，レシピに関するブログ記事を対象とした実験を通して提案手法の有効性を検証した．提案手法は複数あり，単語と HTML タグの系列を既存の固有表現抽出モデルに入力するアプローチの手法と，GNN を用いて HTML 構造情報を読み取るアプローチの手法に分けられる．今後の課題として，どちらのアプローチがどのような特徴を持つ HTML テキストに有効であるかを明らかにするため，様々なブログでデータセットを構築し性能評価実験を行うことが挙げられる．

参考文献

[1] Tetsuya Nakatoh and Sachio Hirokawa. Extraction of Tourist Behavior Contexts from Blog by Verbs and Their Objects. In *Proceedings of the 2012 IIAI International Conference on Advanced Applied Informatics*, 2012.

[2] 齋藤邦子, 鈴木潤, 今村賢治. CRF を用いたブログからの固有表現抽出. 言語処理学会第 13 回年次大会発表論文集, 2007.

[3] 堀達也, 白井清昭. ブログページからのウェブサイト情報・作成者情報の抽出. 言語処理学会第 21 回年次大会発表論文集, 2015.

[4] 池田流弥, 安藤一秋. 深層学習によるブログ記事からの土産の品名・店名抽出. 言語処理学会第 25 回年次大会発表論文集, 2019.

[5] 近江崇宏. Wikipedia を用いた日本語の固有表現抽出のデータセットの構築. 言語処理学会第 27 回年次大会発表論文集, 2021.

[6] Nguyen Tuan Duc, Danushka Bollegala, 石塚満. エンティティペア間類似性を利用した潜在関係検索. 情報処理学会論文誌, Vol. 52, No. 4, pp. 1790–1802, 2011.

[7] 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*, 2019.

[8] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. In *International Conference on Learning Representations*, 2018.

[9] 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*, 2016.

[10] Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. Dice Loss for Data-imbalanced NLP Tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[11] 日本語 Wikipedia エンティティベクトル. `https://github.com/singletongue/WikiEntVec`.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

[13] 日本語 BERT 訓練済みモデル. `https://github.com/cl-tohoku/bert-japanese`.

[14] WHATWG. DOM Living Standard. `https://dom.spec.whatwg.org/`.

[15] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[16] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Neural Information Processing Systems*, 2017.

[17] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the International Conference on Machine Learning*, 2013.

[18] 上辻慶典. アノテーションを支える Orion Annotator の紹介. CyberAgent 秋葉原ラボ 技術報告, Vol. 2, pp. 32–37, 2019.

[19] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2015.

A　実験設定の詳細

ここでは，本文の 4.2 に記載しなかった実験設定の詳細について述べる．

まず，提案手法 1, 2, 3, 4, 5 と HTML タグを利用しない BiLSTM-CRF, BERT の 計 7 つの手法の訓練において共通で，エポック数を 10, バッチサイズを 1, オプティマイザを Adam [19] とした．学習率は，BiLSTM を用いる 2 つの手法では 0.001 とし，BERT を用いる 5 つの手法では 0.00001 とした．また，全ての実験は NVIDIA T4 (16GB) を利用して実行した．

BiLSTM を用いる 2 つの手法においては，ハイパーパラメータとして隠れ状態ベクトルの次元を 10, 50, 100, 150, 200 の中から選択した．

グラフニューラルネットワークを用いる 3 つの提案手法では，レイヤの数と各レイヤの出力次元をハイパーパラメータとしてチューニングした．レイヤの数の候補値は，GCN と GSAGE では 2, 3, 4 層とし，GAT では 2, 3 層とした．出力次元の候補値は 16, 32, 64, 128, 256 とした．なお，組み合わせを減らす目的で，同一モデルの各レイヤの出力次元は同じ値とした．ただし，最後のレイヤの出力次元は固有表現ラベル数で固定した．また，GAT の Multi-head Attention のヘッド数は，2 層のモデルでは順に 4, 6 で固定し，3 層のモデルでは順に 4, 4, 6 で固定した．また，GAT の最終レイヤでは，[8] に倣って式 (6) の CONCAT の部分を平均ベクトルをとる操作に置き換えている．

B　実験結果の詳細

各手法の F1 スコアを記した表 1 に Precision と Recall の評価値を追記したものを表 2 に示す．

表 2　各モデルの F1 スコア, Precision, Recall

	全記事			ブログ A			ブログ B		
モデル	F1	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.
BiLSTM (既存手法)	70.1	77.1	64.3	50.4	59.1	43.9	74.6	74.6	74.6
BiLSTM with Tag	75.3	79.2	71.9	54.4	60.7	49.2	77.6	84.0	72.2
BERT (既存手法)	78.5	78.1	78.9	72.5	71.3	73.7	80.2	83.3	76.9
BERT with Tag	80.2	78.4	82.1	72.6	72.4	72.8	**86.9**	85.6	88.3
BERT + GCN	80.3	78.6	82.1	76.4	76.7	76.0	80.8	81.4	80.2
BERT + GSAGE	80.9	78.8	83.0	76.6	73.1	80.5	80.4	79.5	81.4
BERT + GAT	**81.2**	79.1	83.4	**78.0**	75.8	80.5	81.6	81.7	81.5