文書分類のための クラス情報を考慮したトークン分割

平岡 達也　岩倉 友哉
富士通株式会社
{hiraoka.tatsuya, iwakura.tomoya}@fujitsu.com

概要

本稿では，クラスの情報を有効活用することで文書分類タスクに特化したトークン分割の処理を提案する．ユニグラム言語モデルベースのトークン分割器に，エントロピーに基づく重みを掛け合わせることで，あるクラスに特徴的なトークンを優先的に使用するようなトークン分割を行う．実験より，複数の文書分類のデータセットで提案手法による性能の向上が得られることが分かった．また，クラスが未知の場合であっても，疑似的なクラスによって性能の向上が得られる場合があることを報告する．

1　はじめに

テキストを単語やサブワードなどに区切るトークン分割の処理は，文書分類などの多くの自然言語処理において前処理として取り扱われる．トークン分割は，学習や推論に用いる入力の単位を決定する重要な前処理であり，適切なトークン分割を用いることで自然言語処理の性能の向上が得られる [1, 2, 3]．

適切なトークン分割を行うために，前処理の時点ですでに分かっている後段のタスクの情報を活用できるとよい．例えば，後段のタスクが文書分類である場合，前処理の時点でテキストのドメインや，どのような文書クラスが存在するかが既知であることが多い．従来のトークン分割の手法は後段のタスクの学習に用いるデータで分割器を作成する [4, 5, 6] ため，ドメインに関する情報は考慮することができる．しかしながら，文書分類のクラスなどの後段タスクの学習と推論に用いられる，タスクに固有の情報について考慮することはできない．

本研究では文書分類に焦点を当てて，文書分類のクラスの情報を考慮したトークン分割の新たな手法（Cakenizer: Class-aware tokenizer）を提案する (表 1)．Cakenizer では，文書分類におけるクラスに属するサンプルを一連の文書とみなしてエントロピーに基づくスコアを計算し，クラスに特徴的なトークンが優先的に使用されるような分割を行う．実験結果より，日中英の複数の文書分類のデータセットで性能の向上が得られることを報告する．また，クラスの情報が取得できない場合であっても，K-means クラスタリングによる疑似的なクラスを用いることで，提案手法によるクラス情報を考慮した分割を行い，性能の向上が得られることを報告する．

表 1　学習時に使用する情報の比較．

	使用する情報	テキスト	クラス
従来の NLP	トークン分割器	✓	
	後段モデル	✓	✓
提案手法	トークン分割器	✓	✓
(Cakenizer)	後段モデル	✓	✓

2　言語モデルを用いたトークン分割

あるテキスト S について，$|V|$ 個のトークンから構成される語彙 V のもとで作ることができる S のトークン分割の候補 S' の集合を $V(S)$ と表す．トークン分割の処理では，何らかの方法で計算したトークン分割の確率 $P(S'|S)$ が最大となるような分割候補 $\hat{S}'$ をビタビアルゴリズム [7] などで求める．

$$\hat{S}' = \underset{S' \in V(S)}{\operatorname{argmax}} p(S'|S). \tag{1}$$

ユニグラム言語モデルを用いたトークン分割であれば，$p(S'|S)$ は以下のように計算する．

$$p(S'|S) = \prod_{s \in S'} p(s). \tag{2}$$

ここで s は S' を構成する $|S'|$ 個のトークンのひとつであり，$S' = \langle s_1, \ldots, s_{|S'|} \rangle$ である．ユニグラムの確率 $p(s)$ の推定には，EM アルゴリズムを用いた学習 [8] などを利用する．

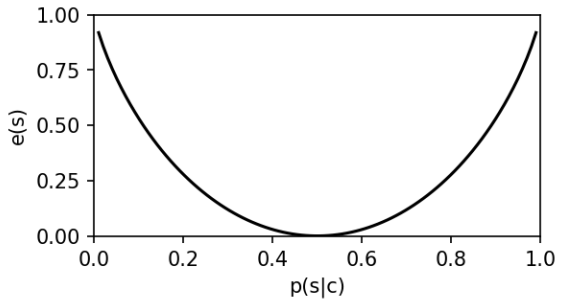

図 1　二値分類での $p(s|c)$ に対するスコア $e(s)$ の値域.

3　クラスを考慮したトークン分割

提案手法では，後段タスクの文書分類で使用するクラス $c \in C$ に関する情報を盛り込んだトークン分割を行う．クラスの情報を考慮したトークン分割を行うために，式 (2) におけるトークンの確率 $p(s)$ を以下の μ による重み付き和 $p'(s)$ で置き換える．

$$p'(s) = (1-\mu)p(s) + \mu q(s). \tag{3}$$

$q(s)$ は，トークン s の各クラスでの分布に対するエントロピーをもとに計算したスコア $e(s)$ によって重みづけられた $p(s)$ である．

$$q(s) = \frac{p(s)e(s)+\epsilon}{\sum_{t\in V} p(t)e(t)+\epsilon}, \tag{4}$$

$$e(s) = 1 + \sum_{c\in C} p(s|c)\log_{|C|} p(s|c). \tag{5}$$

ここで ϵ は，トークンの確率が 0 になることを防ぐための微小な値である．また，語彙に含まれるすべてのトークンに対応する $q(s)$ の和は 1 となる．$|C|$ は，文書分類のタスクで使用されるクラスの総数であり，$p(s|c)$ はクラス c に分類されるテキストの集合においてトークン s が出現する条件付確率である．なお，$p(s|c) = 0$ とならないように微小な値を加えたうえで計算を行う．

図 1 に，二値分類において s のクラスごとの分布の割合に対して重み $e(s)$ がとる値域を示した．トークン s が特定のクラス（例えば正負クラスのどちらか一方）にしか出現しない場合，$e(s) \fallingdotseq 1.0$ であり，s がどのクラスにも等しく出現する場合は $e(s) = 0.0$ となる．すなわち $e(s)$ は，特定のクラスに偏って出現する特徴的なトークンほど重要度を高め，様々なクラスに出現する普遍的なトークンほど重要度を低くするような重みとして働く．これにより，特定のラベルに特徴的なトークンを優先して使用するようなトークン分割が得られると期待される．

Algorithm 1 提案手法を Viterbi-Training に組み込む例

Require: 学習データ D，最終的な語彙の規模 K，内部学習ループ数 M，語彙の削減率 R，係数 μ.

1: D に出現する文字列の集合で語彙 V を初期化
2: **while** 語彙の規模 $|V| > K$ **do**
3:　　$p(s) \in \theta_V \leftarrow$ ランダム初期化
4:　　**for** $m = 1$ to M **do**
5:　　　　$D'_{\theta_V} \leftarrow D$ を θ_V で 1-best のトークン分割
6:　　　　$q(s) \in \mathbf{W}_V \leftarrow D'_{\theta_V}$ に基づいて重みを計算
7:　　　　$\hat{\theta_V} \leftarrow (1-\mu)\theta_V + \mu\mathbf{W}_V$ （式 (3) に対応）
8:　　　　$D'_{\hat{\theta_V}} \leftarrow D$ を $\hat{\theta_V}$ で 1-best のトークン分割
9:　　　　$\theta_V \leftarrow D'_{\hat{\theta_V}}$ で数え上げたユニグラム確率
10:　　　　θ_V の下位 $|V| \times R$ トークンを V から削除
11: $\theta_V \leftarrow D$ と V から 4–9 行目と同様に再推定

トークンの確率 $p(s)$ を学習済みのユニグラム言語モデルに対して，後から重みの項 $q(s)$ を付け加えることで提案手法は実装できる．また，SentencePiece [5] の Unigram モードのように，各トークンの確率を推定したり語彙の枝刈りを行うトークン分割器の作成の段階から，重みを用いて学習することもできる．Viterbi-Training [8] を用いたトークンの確率の推定と語彙の枝刈りを行う例を，Algorithm 1 に示した．アルゴリズムにおいて，5–7 行目の重みづけをスキップすれば，従来のユニグラム言語モデルの確率の推定と語彙の枝刈りと同様の処理になる．また，EM-Training [8] では，8 行目と 9 行目のパラメータの推定に EM アルゴリズムを用いる．

4　実験: オリジナルのクラス

提案手法（Cakenizer）の効果を，日本語・中国語・英語の 3 言語での文書分類のタスクを用いて検証する．本節では，文書分類のデータセットに設定されている本来のクラスのラベルを用いて，トークン分割器の学習を行う設定について実験する．

4.1　設定

4.1.1　データセット

日本語の文書分類では，Twitter の感情分類のデータセット [9, 10] を用いた．また，JGLUE [11] より JNLI と JMARC を用いた．中国語では，Weibo の感情分類のデータセット [12] を用いた．また，EC サイト（JD.com）のレビューデータセット [13] からレート予測とジャンル予測の文書分類タスクを作成

表 2　文書分類のタスクでの実験結果（F1% ± SD）．太字は Vanilla の値を超える結果．

		Viterbi-Training			EM-Training		
言語	データセット	Vanilla	PostProc	Dynamic	Vanilla	PostProc	Dynamic
日本語	WRIME	$41.26_{\pm 0.91}$	**41.88**$_{\pm 0.65}$	**42.31**$_{\pm 0.80}$	$40.75_{\pm 0.82}$	**40.86**$_{\pm 1.04}$	**40.88**$_{\pm 0.84}$
	Twitter	$84.91_{\pm 0.18}$	**84.95**$_{\pm 0.16}$	$84.90_{\pm 0.22}$	$84.74_{\pm 0.16}$	$84.71_{\pm 0.25}$	$84.69_{\pm 0.19}$
	JNLI	$49.46_{\pm 1.24}$	**49.95**$_{\pm 1.01}$	**50.60**$_{\pm 1.71}$	$49.85_{\pm 0.60}$	**50.13**$_{\pm 1.7}$	**50.23**$_{\pm 0.63}$
	MARC	$91.74_{\pm 0.45}$	**92.14**$_{\pm 0.34}$	**92.01**$_{\pm 0.38}$	$91.91_{\pm 0.56}$	$91.84_{\pm 0.39}$	**92.08**$_{\pm 0.35}$
中国語	Weibo	$92.61_{\pm 0.10}$	$92.60_{\pm 0.08}$	**92.67**$_{\pm 0.10}$	$92.60_{\pm 0.09}$	$92.54_{\pm 0.07}$	**92.59**$_{\pm 0.07}$
	JD.com-Rating	$47.39_{\pm 0.17}$	**47.53**$_{\pm 0.16}$	**47.48**$_{\pm 0.20}$	$47.30_{\pm 0.15}$	$47.19_{\pm 0.32}$	$47.14_{\pm 0.35}$
	JD.com-Genre	$46.43_{\pm 0.18}$	**46.54**$_{\pm 0.16}$	**46.54**$_{\pm 0.24}$	$46.33_{\pm 0.24}$	$46.31_{\pm 0.15}$	**46.47**$_{\pm 0.14}$
英語	Twitter	$71.60_{\pm 0.65}$	$71.41_{\pm 0.55}$	$71.28_{\pm 0.29}$	$71.74_{\pm 0.64}$	**71.83**$_{\pm 0.30}$	**71.79**$_{\pm 0.59}$
	Amazon-Rating	$61.66_{\pm 0.35}$	**61.68**$_{\pm 0.41}$	$61.35_{\pm 0.49}$	$61.73_{\pm 0.38}$	**62.10**$_{\pm 0.41}$	**61.91**$_{\pm 0.55}$
	Amazon-Genre	$61.99_{\pm 0.36}$	$62.24_{\pm 0.32}$	$61.88_{\pm 0.45}$	$62.30_{\pm 0.58}$	**62.35**$_{\pm 0.47}$	**62.36**$_{\pm 0.30}$
	Average	$64.91_{\pm 0.18}$	**65.09**$_{\pm 0.15}$	**65.10**$_{\pm 0.21}$	$64.91_{\pm 0.15}$	**64.99**$_{\pm 0.18}$	**65.01**$_{\pm 0.15}$

した [14]．英語では，Twitter の感情分類データセットを用いた [15]．また，EC サイト（Amazon）のレビューデータセット [16] からレート予測とジャンル予測の文書分類タスクを作成した．

4.1.2　トークン分割器

実験では，提案手法による重みづけを用いない従来の方法でユニグラム言語モデルの確率の推定と語彙の枝刈りを行う手法（**Vanilla**）をベースラインとして，ユニグラム言語モデルの確率の推定と語彙の枝刈りの段階から重みを使用する場合（**Dynamic**）を比較する．さらに，提案手法の単純な適用方法として，Vanilla に対して後処理として式 (3) の重みを付与した手法（**PostProc**）を比較する．

全ての実験において語彙の規模は $K = 16,000$ であり，トークン分割器の作成には各データセットの学習データを D として使用した．Algorithm 1 の 1 行目における語彙の初期化では，頻度が 4 以下のトークンと長さが 9 以上のトークンは除外した．語彙の枝刈りイテレーションにおける，ユニグラム言語モデルの確率の推定ループの回数は $M = 2$，イテレーションごとの語彙の削減率は $R = 0.4$ とした．また，重み付き和の係数は $\mu = 0.5$ とした．

4.2　文書分類器

トークン分割器の差による影響を観察しやすいように，実験では 1 層の BiLSTM と 3 層の MLP を用いた単純な文書分類器 [17] を用いた．分類器は 20 エポックの学習を行い，Adam [18] を用いて最適化した．検証データで性能が最大となったチェックポイントを採用し，評価データでの F1 値を計測した．

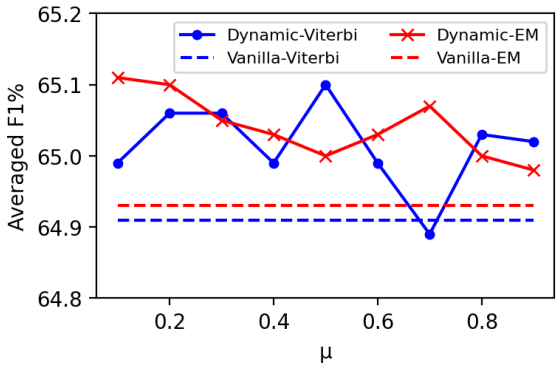

図 2　μ が性能に与える影響．

本実験においては，トークン分割器の作成と文書分類器の学習の双方に確率的な要素が含まれている．そのため，トークン分割器の作成と文書分類器の学習をそれぞれ 3 回ずつ組み合わせて試行し，その平均の F1 値を報告する．すなわち，それぞれパラメータをランダムに初期化した 3 つのトークン分割器と，3 つの文書分類器の合計 9 つの試行パターンについて平均を取った．

4.3　結果

表 2 にまとめた実験結果より，提案手法は Viterbi/EM-Training の双方の設定において，複数のデータセットでの性能向上に寄与しうることが示唆された．最終的な平均 F1 値は Viterbi-Training の学習時に提案手法を用いる設定が最も高く，Cakenizer は Viterbi-Training と組み合わせて使うこと良いと言える．また，提案手法を後処理として適用する方法も，学習時から組み込む方法と同様に性能向上に寄

表 3　疑似ラベルを用いた実験の結果．太字は表 2 の Vanilla を超える結果．

		Viterbi-training						EM-training					
		PostProc			Dynamic			PostProc			Dynamic		
	データセット	L=5	L=10	L=15	L=5	L=10	L=15	L=5	L=10	L=15	L=5	L=10	L=15
日本語	WRIME	**41.97**	**41.65**	**41.42**	**42.13**	**41.49**	**41.67**	**41.13**	**40.76**	40.34	**41.38**	**41.10**	**41.02**
	Twitter	**85.01**	**84.92**	**84.94**	**84.98**	84.87	84.90	84.65	84.67	84.74	84.74	84.63	84.71
	JNLI	**49.83**	49.37	**50.25**	**49.53**	**50.02**	**50.49**	49.47	**50.97**	**50.70**	**50.29**	**50.77**	**49.94**
	MARC	**92.23**	**91.88**	**92.24**	**92.14**	**92.22**	**92.10**	91.88	91.55	**92.03**	**92.11**	**92.07**	**92.16**
中国語	Weibo	**92.62**	**92.64**	92.61	92.61	**92.67**	92.56	92.59	92.60	92.58	92.57	92.60	**92.62**
	JD.com-Rating	47.29	**47.47**	**47.51**	**47.42**	**47.48**	**47.49**	**47.33**	47.25	47.12	47.17	47.30	**47.38**
	JD.com-Genre	**46.45**	**46.50**	**46.63**	**46.50**	**46.57**	**46.47**	**46.39**	**46.38**	**46.37**	**46.39**	46.33	**46.41**
英語	Twitter	71.53	**71.69**	**71.83**	71.38	**71.71**	**71.70**	**71.82**	71.36	**71.75**	71.35	71.51	71.57
	Amazon-Rating	61.55	**61.72**	**61.83**	**61.69**	**61.82**	**61.70**	**61.84**	**61.83**	**61.78**	**61.84**	61.64	61.49
	Amazon-Genre	**62.49**	**62.14**	**62.40**	61.65	**62.29**	**62.09**	62.19	62.17	62.30	62.28	62.30	**62.41**
	Average	**65.10**	**65.00**	**65.17**	**65.00**	**65.11**	**65.12**	**64.93**	**64.95**	**64.97**	**65.01**	**65.03**	**64.97**

与することが示唆された．後処理として提案手法を用いる方法は導入のコストが低いため，多くの場面での活用が期待できる．

図 2 に，Dynamic の設定で式 (3) での重み付き和の係数 μ を 0.1 から 0.9 の範囲で変化させたときの平均 F1 値の推移を示した．全体的にベースラインである Vanilla を超える値で推移しているが，μ が大きくなるにつれてわずかに向上幅が下がる傾向がみられる．また，EM-Training に提案手法を組み込んだ場合の方が，異なる μ に対して安定して性能の向上が得られることが示唆された．

5　実験: 疑似ラベル

実践的な自然言語処理では，トークン分割を行う段階で本来の分類ラベルが利用できない場合がある．本節では，学習データに対して付与した疑似ラベルを用いて提案手法のトークン分割器を作成した場合の効果について検証する．

5.1　設定

疑似ラベルを作成するために，K-means クラスタリングを各文書分類のデータセットの学習データに適用した．クラスタリングを行う段階ではトークン分割器が使用できないため，テキストの特徴量の作成にはテキストに含まれる文字 N-gram ($N = \{1, 2, 3, 4, 5\}$) を用いた．scikit-learn に実装されている K-means クラスタリング [19] を利用し，クラスタの数 $L = 5, 10, 15$ について実験を行った．疑似ラベルを用いる点を除けば，全ての実験設定は §4 と同じである．

5.2　結果

表 3 に実験結果より，提案手法によるトークン分割に疑似ラベルを用いた場合であっても，Vanilla に比べて性能の向上が得られることが分かった．また，設定の組み合わせによっては，疑似ラベルを用いた場合の方がオリジナルのラベルを用いた場合よりも高い性能向上が得られることが分かった．これは，教師なし学習によって獲得した疑似ラベルは機械学習のモデルにとって扱いやすいドメインの割り振りになっており，このドメインを考慮したトークン分割を行うことで効率的な文書分類の学習ができるためであると考えられる．この結果は，文書分類タスクにおいて疑似的なラベルを用いて事前学習を行うことで性能向上が得られるという既存研究の報告 [20] にも合致する．

6　おわりに

本稿では，文書分類に特化したトークン分割の手法として Cakenizer を提案した．提案手法はユニグラム言語モデルをベースとし，トークンの確率の推定と語彙の枝刈りの過程で後段の文書分類のラベルを考慮する．実験の結果より，日本語・中国語・英語での文書分類で性能向上に寄与することが示唆された．また，疑似ラベルを用いた設定でも性能の向上が得られることが分かった．今後は機械翻訳などの文書分類以外のタスクのデータセットに対しても，疑似ラベルを用いた手法の応用を行う．

謝辞

本研究は，JST，ACT-X，JPMJAX21AM の支援を受けたものです．

参考文献

[1] Tatsuya Hiraoka, Hiroyuki Shindo, and Yuji Matsumoto. Stochastic tokenization with a language model for neural text classification. In **Proceedings of the 57th Annual Meeting of ACL**, pp. 1620–1629, 2019.

[2] Kaj Bostrom and Greg Durrett. Byte pair encoding is suboptimal for language model pretraining. In **Findings of ACL: EMNLP 2020**, pp. 4617–4624, 2020.

[3] Hoo-Chang Shin, Yang Zhang, Evelina Bakhturina, Raul Puri, Mostofa Patwary, Mohammad Shoeybi, and Raghav Mani. BioMegatron: Larger biomedical domain language model. In **Proceedings of the 2020 Conference on EMNLP**, pp. 4700–4706, Online, November 2020. Association for Computational Linguistics.

[4] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In **Proceedings of the 54th Annual Meeting of ACL (Volume 1: Long Papers)**, Vol. 1, pp. P1715–1725, 2016.

[5] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In **Proceedings of the 2018 Conference on EMNLP: System Demonstrations**, pp. 66–71, 2018.

[6] Xinying Song, Alex Salcianu, Yang Song, Dave Dopson, and Denny Zhou. Fast wordpiece tokenization. In **Proceedings of the 2021 Conference on EMNLP**, pp. 2089–2103, 2021.

[7] Andrew Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. **IEEE transactions on Information Theory**, Vol. 13, No. 2, pp. 260–269, 1967.

[8] Sabine Deligne and Frederic Bimbot. Language modeling by variable length sequences: Theoretical formulation and evaluation of multigrams. In **1995 International Conference on Acoustics, Speech, and Signal Processing**, Vol. 1, pp. 169–172. IEEE, 1995.

[9] Yu Suzuki. Filtering method for twitter streaming data using human-in-the-loop machine learning. **Journal of Information Processing**, Vol. 27, pp. 404–410, 2019.

[10] Tomoyuki Kajiwara, Chenhui Chu, Noriko Takemura, Yuta Nakashima, and Hajime Nagahara. WRIME: A new dataset for emotional intensity estimation with subjective and objective annotations. In **Proceedings of the 2021 Conference of NAACL: HLT**, pp. 2095–2104, Online, June 2021. Association for Computational Linguistics.

[11] Kentaro Kurihara, Daisuke Kawahara, and Tomohide Shibata. JGLUE: Japanese general language understanding evaluation. In **Proceedings of the Thirteenth LREC**, pp. 2957–2966, Marseille, France, June 2022. European Language Resources Association.

[12] S2ap: A sequential senti-weibo analysis platform. https://github.com/wansho/senti-weibo.

[13] Yongfeng Zhang, Min Zhang, Yi Zhang, Guokun Lai, Yiqun Liu, Honghui Zhang, and Shaoping Ma. Daily-aware personalized recommendation based on feature-level time series analysis. In **Proceedings of the 24th international conference on WWW**, pp. 1373–1383, 2015.

[14] Tatsuya Hiraoka, Sho Takase, Kei Uchiumi, Atsushi Keyaki, and Naoaki Okazaki. Joint optimization of tokenization and downstream model. In **Findings of ACL: ACL-IJCNLP 2021**, pp. 244–255, 2021.

[15] Twitter sentiment analysis. https://www.kaggle.com/c/twitter-sentiment-analysis2.

[16] Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In **proceedings of the 25th international conference on WWW**, pp. 507–517, 2016.

[17] Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. Text classification improved by integrating bidirectional lstm with two-dimensional max pooling. In **Proceedings of COLING 2016, the 26th COLING: Technical Papers**, pp. 3485–3495, 2016.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. **arXiv preprint arXiv:1412.6980**, 2014.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, Vol. 12, pp. 2825–2830, 2011.

[20] Eyal Shnarch, Ariel Gera, Alon Halfon, Lena Dankin, Leshem Choshen, Ranit Aharonov, and Noam Slonim. Cluster & tune: Boost cold start performance in text classification. In **Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 7639–7653, 2022.