コミュニティ特有のハッシュタグ空間を考慮したマルチラベル候補生成器を用いる二段階推薦

深澤祐援
コミューン株式会社
fukasawa.yusuke@commune.co.jp

概要

近年, 顧客とのエンゲージメントを生む施策としてコミュニティの重要性や価値が増しており, コミューン株式会社が提供する “Commune” のようにクライアントごとに個別のコミュニティを提供するサービスが注目されている. このようなコミュニティ提供は顧客体験の向上に寄与する一方で, 例としてハッシュタグ推薦タスクにおいては, 各コミュニティが異なるハッシュタグ空間を持つため新たな課題をもたらす. 本研究では, 単一のモデルで複数のコミュニティに対するハッシュタグ推薦を可能にするべくマルチラベル分類モデルを用いた二段階推薦を提案する. 提案手法は実験の結果, NDCG@10 で 0.52 を達成し、推論速度の観点からも実運用に適した有効性を確認した.

1 はじめに

様々な SNS サービスが存在する昨今において, 顧客との密接な関係性を構築することなどを目的としてクライアントごとのコミュニティを提供するサービスが登場している. 例としてコミューン株式会社が提供する “Commune” が挙げられる. 個別のコミュニティを提供することは顧客体験の充実に繋がる可能性がある一方で, 機械学習を用いた施策を適用する上では問題点も生じる.

本研究ではその対象としてハッシュタグを推薦するタスクを考える. ユーザ全員が利用できるハッシュタグが共通ならば一般的な推薦アルゴリズムによって解決できる. 一方でそれぞれのハッシュタグが別々のコミュニティ空間に閉じている場合, あるコミュニティに対して別のコミュニティのハッシュタグを推薦することはできない. サービス成長に伴うコミュニティ数の増加を考慮した場合, 各コミュニティごとにハッシュタグ推薦モデルを構築する場合は増え続けるモデルリソースを管理する必要があり, 最適なアプローチではない.

そこで本研究では, 単一の推薦モデルを構築しつつ別々のコミュニティに対するハッシュタグ推薦を行う方法を提案する. 投稿テキストを入力としたモデルが候補生成時にコミュニティごとの推薦候補を出力した後にリランキングを行う二段階推薦の枠組みを取った手法である. 本研究ではコミューン株式会社が保有するデータを用いた実験を行い, 提案手法の有効性を確認する.

2 関連研究

二段階推薦のアプローチは Youtube の動画推薦問題に対して提案した Paul ら [1] が先行事例として存在する. 近年では候補生成を別のモデルで行い LLM によるリランキングを行う手法 [2] も提案されている.

SNS サービスを想定してハッシュタグを推薦する研究はこれまで盛んに行われてきている. Godin ら [3] は Twitter におけるハッシュタグについて LDA を用いたトピックモデリングによって推薦を行う手法を提案している. Gong ら [4] は Attention と CNN を組み合わせた手法を提案した.

ハッシュタグ推薦問題に対して sigmoid を用いるマルチラベル分類でアプローチしている研究としては Bansal ら [5] の研究が挙げられる.

ハッシュタグ推薦の先行事例の多くが全ユーザが共通のハッシュタグを利用できる状況を想定している. 本研究の重要な点として対象とするコミュニティごとに利用できるハッシュタグが異なる状況に対するアプローチを提案している点にある.

3 手法: 二段階推薦

本研究ではハッシュタグ推薦の手法として二段階推薦の枠組みを取る. それぞれのコミュニティに

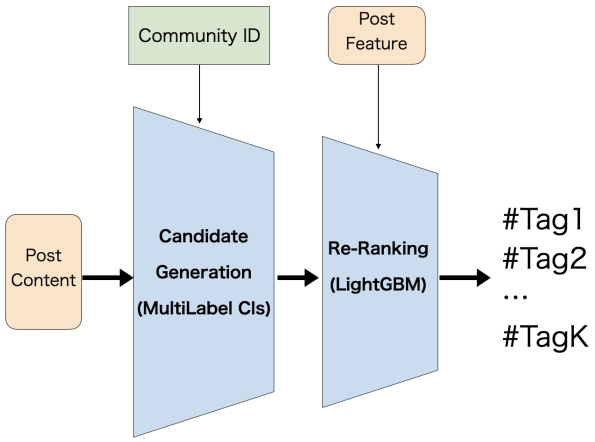

図 1　本研究で提案するハッシュタグ推薦モデル

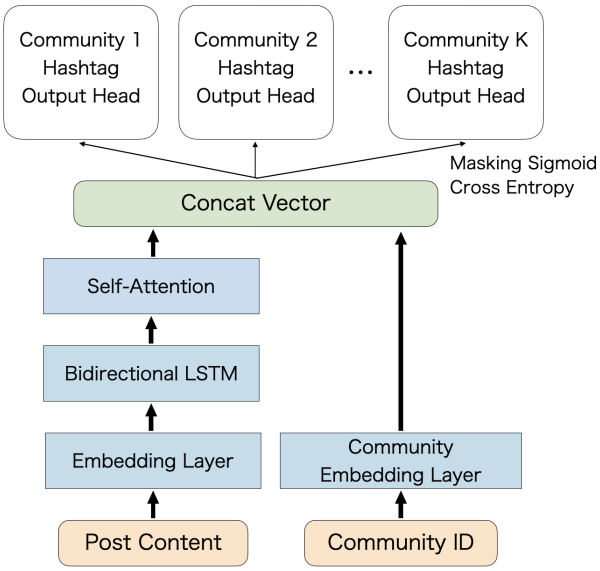

図 2　候補生成器として用いるマルチラベル分類モデル

よって用いることができるハッシュタグが異なるという問題に対して, コミュニティごとの特徴を考慮した単一のマルチラベル分類器を候補生成時に用いることを提案する. そして得られた候補に対して LightGBM によるスコアリングを行い最終的な推薦結果を獲得する. 図 1 に概要を示す.

3.1　候補生成: マルチラベル分類

本研究ではマルチラベル分類モデルを候補生成器として扱う. 図 2 にその概要を示す. モデル内での計算は次の順序で行われる.

1. 投稿テキストに対して形態素解析を行い単語ごとに分割する
2. Embedding Layer でベクトルに変換する
3. Bidirectional LSTM, Self-Attention 層を通して変換する
4. 各投稿に紐づくコミュニティ ID を Community Embedding Layer でベクトルに変換する
5. Community Embedding と投稿テキストベクトルを Concat する
6. コミュニティごとの出力層を用意しておき, 投稿ごとに対応する出力層からハッシュタグ推薦候補を取得する

3.1.1　推薦候補の出力と損失計算

損失計算を行う際, モデルが出力する logits はコミュニティに対応したハッシュタグ数に対応しているが, 正解ラベル側はバッチ内で固定長の長さになっている必要がある. そのため, 正解ラベルにおける当該コミュニティとは関連しないハッシュタグ候補は PAD token とする. これらを無視して正解ラベルとの Binary Cross Entropy 損失を計算するため, マスキング付きの BCE 損失関数を用いる.

3.2　ランキング: LightGBM

前述した候補生成モデルにより推薦候補を出力した後, リランキングのためのスコア付与を行う. ここでは LightGBM [6] を用いる. LightGBM には投稿テキストとハッシュタグテキスト及び特徴量を入力し, その投稿に対してハッシュタグが付与されるスコアの予測を行う. 学習時には予めその投稿に紐付くハッシュタグではないタグをランダムサンプリングによって同数抽出し, ネガティブとして与える. この過程を通じて当該投稿に対してポジティブ, ネガティブなハッシュタグのデータを用意し, 二値分類問題として LightGBM の学習を行う.

4　結果と考察

コミューン株式会社が運営するプラットフォーム上での 2024 年 9 月から 12 月のデータを対象に実験を行った. なお当該期間における投稿件数が 10 件以下のコミュニティは除外した.

4.1　事前設定

データは予め 80 %を学習データ, 残りをテストデータとして学習データを用いて候補生成モデル・ランキングモデルを学習させた後, テストデータに対してモデルは 10 件のハッシュタグを推薦する. Precision, Recall, MAP, MRR, NDCG による比較を行った. 各モデルで用いる埋め込みモデルには fastText に対して Bloom Embedding などの拡張を

表 1　各モデルにおける評価結果

モデル	Precision@10	Recall@10	MAP@10	MRR@10	NDCG@10
Simple Candidate Generator	0.0187	0.0704	0.0283	0.0572	0.0468
MultiLabel Classifier	0.1590	0.5805	0.3838	0.5417	0.4773
LightGBM	0.1624	0.6028	0.3528	0.4988	0.4601
LightGBM + Simple Candidate Generator	0.1699	0.6274	0.3905	0.5411	0.4941
LightGBM + MultiLabel Classifier	**0.1812**	**0.6716**	**0.4229**	**0.5706**	**0.5292**

行った埋め込みモデルである floret [7] を用いた. 予め dim=300, skip-gram の設定でコミューンが保持する投稿テキストデータに対する事前学習を行った. 投稿内のテキスト及びハッシュタグを単語分割した後, floret を用いてベクトル化し連結させたものを特徴量とした.

4.2　ベースライン手法

本研究で対象とした手法である LightGBM とマルチラベル分類器による二段階推薦に加えて, それぞれのモデルを単一で用いて推薦を行う手法, また Simple Candidate Generator を用いて推薦を行う場合との比較を行った.

Simple Candidate Generator　投稿テキスト内の単語との編集距離・コミュニティにおける出現頻度・投稿テキストとハッシュタグを同じモデルで文書ベクトルに変換し, 投稿テキストベクトルに対しての近傍, という 3 種類の方策で得られたハッシュタグを候補として出力する. 近傍探索数は 30 とした.

4.3　実験結果

表 1 に実験結果を示す. まず結果から述べられることとして, 二段階推薦の優位性が確認できたことが挙げられる. LightGBM 及びマルチラベル分類器のみを使って推薦する場合よりも候補生成器と組み合わせた場合のほうが良い結果となった.

またマルチラベル分類器を候補生成器として用いることで Simple Candidate Generator を使う場合よりも すべての指標で良好な結果が得られている. 実際に投稿テキストに対して付与されたハッシュタグのデータを用いて学習を行ったことで効果的な候補生成が可能になることがわかった.

4.4　エラー分析

ここでは候補生成器として Simple Candidate Generator とマルチラベル分類モデルによる候補生成結果の違いについて述べる. 表 2 に出力結果の一例として上位 10 件の候補生成結果を示す. 実際のタグとマルチラベル分類モデルによる生成結果を比較するとこちらは 4 件のタグが予測が成功している. 一方で Simple Candidate Generator では 2 件に留まっている. この中でも “新機能” というタグについて着目すると, 投稿内容の中にこの単語が含まれているが後者の結果には含まれていない. これは分かち書きの問題によるものである. “新機能” は UniDic 辞書もしくは IPADic 辞書を用いた MeCab による処理では “新・機能” と分かち書きされる. Simple Candidate Generator では編集距離によるスコアリングも行われているものの他のタグがより高いスコアを記録したため上位 10 件には “新機能” タグが出現しなかったと考えられる. こうした事象の積み重ねによって候補生成の質に差が生じ, 推薦評価結果に現れたといえる.

また図 3 で示されているように, NDCG@10 が高く推薦が機能しているコミュニティは平均投稿数が

表 2　投稿データおよび各候補生成結果の比較例

投稿内容	コミュマネ CS のみなさん こんにちは。新機能を解説するイベントをやってほしいのお声から新機能リリース説明会 (オンライン Zoom ウェビナー) の開催が決定！ぜひご参加しませんか？？

候補生成モデル	生成結果 (上位 10 件)
マルチラベル	**お知らせ**, **コミュニティ運営**, **イベント**, 悩んでます, コミュニティ知見, はじめての方へ, オンラインイベント, **新機能**, DM, 関西
シンプル	チャレンジ, コミュニティ運営, 悩んでます, ウェビナー, **解説**, リリース, 仲間, オンラインイベント, **イベント**, コミュニティ
実際のタグ	新機能, 解説, はじめての方へ, リリース, お知らせ, コミュニティ運営, イベント

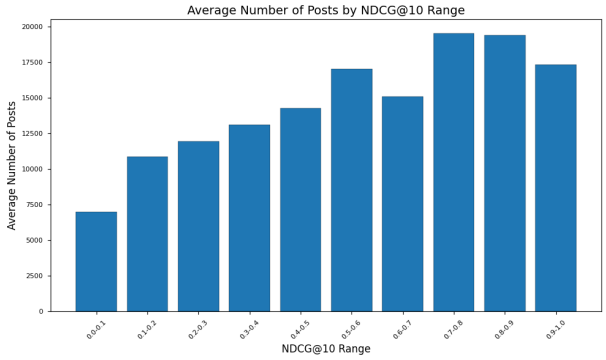

図 3　コミュニティの平均投稿数に応じた NDCG@10 の比較

そうでない場合と比べて多いことが確認された. この結果は, 投稿数の違いが推薦精度に影響を与える可能性を示唆している. 特に比較的データ量が少ないコミュニティに対する推薦精度の向上が今後の課題であると考えられる.

4.5　埋め込みモデルについての比較

今回用いた floret とは別に LightGBM とマルチラベル分類モデルを組み合わせた二段階推薦について, 埋め込みモデルに関する手法の比較を行った.

4.5.1　TF-IDF

投稿内のテキスト及びハッシュタグを単語分割した後, それぞれを TF-IDF によるベクトル化を行って連結させる. 更にそれを TruncatedSVD を用いて 300 次元に圧縮したものを LightGBM の特徴量とする.

4.5.2　Ruri

日本語性能を高めた埋め込みモデルである Ruri[8] (cl-nagoya/ruri-small) を比較対象とした. Ruri を用いる上では二つの方式を試した. まず一つが TF-IDF, floret と同様に投稿テキストとハッシュタグをそれぞれベクトル化した後に連結させる方策 (Separate) である. Ruri の学習時の設定に倣い, ハッシュタグには “クエリ: ”, 投稿テキストには “文章: ” を prefix として設定した. もう一つが SEP トークンで投稿テキストとハッシュタグを結合させて一つの文章とし, それをベクトル化する方策 (Unified) である. こちらは結合した文章に対して “文章: ” を prefix として設定した.

表 3　LightGBM で用いる Vectorizer についての比較結果

Vectorizer	NDCG@10	変換速度
TF-IDF with SVD	0.4550	0.0253s/it
floret	**0.5292**	**0.0001s/it**
cl-nagoya/ruri-small		0.0788s/it
with unified	0.1995	-
with separate	0.4785	-

4.5.3　結果

得られた結果のうち NDCG@10 を表 3 に示す. 今回の手法の中で最も高い精度だったのは floret を用いる場合となった. 次点で Ruri を Separate で用いる方策であった. Separate が優れていた理由として, ハッシュタグ推薦の問題において文脈情報がそれほど重要でなかった可能性が考えられる. 文字情報だけでは紐付きが見出だせないハッシュタグがつくケースも多いが, それ以上に表層的な文字情報を直接活かして推論に役立てることが求められていたと思われる.

また各モデルについて, 400 文字程度のテキストに対するベクトルへの変換速度を測定した. Ruri を用いる場合の変換速度は平均約 0.07 秒 (CPU にて) であった. 実運用を見据えた推論速度を考えても floret を用いる場合が最も適切なモデルであったと言える.

5　おわりに

本研究ではハッシュタグの空間が異なる別々のコミュニティに対して, 候補生成器にマルチラベル分類モデルを用いた二段階推薦によってハッシュタグ推薦を行う手法を提案した. 実験の結果, NDCG@10 で 0.52 及び推論速度という実運用を見据えた観点から見ても提案手法が有効であることが確認された. 特に, 単一のマルチラベル分類モデルであっても複数の出力層を持たせて学習させることでコミュニティごとに有効な候補を生成できることがわかった.

一方で, 学習データが相対的に少ないコミュニティに対しては推薦が上手く機能していない可能性が示唆された. 推薦が容易でないコミュニティの推薦に対して損失関数を動的に調整するなどの改善が必要だと考えられる.

参考文献

[1] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In **Proceedings of the 10th ACM Conference on Recommender Systems**, New York, NY, USA, 2016.

[2] Zhenrui Yue, Sara Rabhi, Gabriel De Souza Pereira Moreira, Dong Wang, and Even Oldridge. Llamarec: Two-stage recommendation using large language models for ranking. **ArXiv**, Vol. abs/2311.02089, , 2023.

[3] Fréderic Godin, Viktor Slavkovikj, Wesley De Neve, Benjamin Schrauwen, and Rik Van de Walle. Using topic models for twitter hashtag recommendation. **Proceedings of the 22nd International Conference on World Wide Web**, 2013.

[4] Yu Han Gong and Qi Zhang. Hashtag recommendation using attention-based convolutional neural network. In **International Joint Conference on Artificial Intelligence**, 2016.

[5] Shubhi Bansal, Kushaan Gowda, and Nagendra Kumar. A hybrid deep neural network for multimodal personalized hashtag recommendation. **IEEE Transactions on Computational Social Systems**, Vol. 10, pp. 2439–2459, 2023.

[6] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: a highly efficient gradient boosting decision tree. In **Proceedings of the 31st International Conference on Neural Information Processing Systems**, NIPS'17, p. 3149–3157, Red Hook, NY, USA, 2017. Curran Associates Inc.

[7] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. **Transactions of the Association for Computational Linguistics**, Vol. 5, pp. 135–146, 2017.

[8] Hayato Tsukagoshi and Ryohei Sasano. Ruri: Japanese General Text Embeddings, 2024.