教師なし分割と言い換えに基づく化合物名同一性判定における候補絞り込み

浦澤 合† 乾 孝司† 田中一成§ 岩倉友哉§

†筑波大学大学院/理研 AIP-富士通連携センター
§富士通研究所/理研 AIP-富士通連携センター

代表連絡先：g.u@mibel.cs.tsukuba.ac.jp

1 はじめに

化学分野では、化合物の名称とその属性等をまとめた化合物 DB が整備され、研究開発に利用される。このとき、ある化合物の名称には別称が多く存在するため、化合物 DB を整備する際は、同一の化合物をあらわす名称をまとめあげるための同一性判定処理が必要不可欠となる。しかしながら、一般に化合物 DB は非常に大規模であるため、総当り的に同一性を判定することは処理コストの面で大きな負担となる。

この問題を回避する一つの手段として判定候補の絞り込みがある。判定候補を準備することによって、ある化合物名に対し、それと同一の化合物をあらわす名称を総当りで確認するのではなく、より小規模な候補のみを判定対象することが可能になる。本研究では、上記背景に基づき、化合物名の分割処理と言い換え処理に基づく候補絞り込み手法を提案し、その有効性を検証する。

2 問題設定

本研究では、判定候補の絞り込み処理を以下のような検索問題として考える。

- 入力
  - 検索クエリ：ある化合物名
  - 検索元データ：大規模化合物名集合
- 出力
  - クエリと同一化合物をあらわす化合物名

検索の目的は、検索元データから入力化合物名と同一の化合物をあらわす化合物名をもれなく含む（小規模な）検索結果を得ることである。検索クエリが検索元データから見て別称となる状況を想定するため、検索元データには、検索クエリと同一名称ではないが検索クエリと同一の化合物をあらわす化合物名が 1 件以上含まれているものとする。また、検索結果は順位付きで出力され、その上位 N 件（N-best）を使い、検索性能を評価するものとする。

この検索問題の設定では、単純に検索クエリとなる化合物名そのもので検索を試みると 1 件も検索できない。本研究ではこの検索問題に対して、検索目的を達成するために、検索クエリの分割処理および言い換え処理に基づくクエリ拡張をおこなうことを考え、それらの手法を提案する。

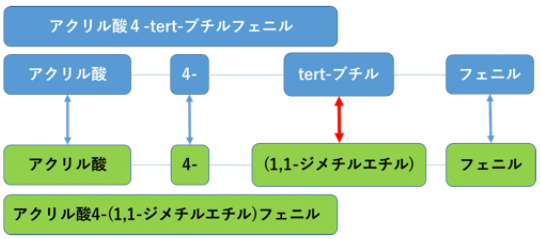

図 1: サブワードクエリ：同一化合物をあらわす化合物名は部分的に共通要素をもつ傾向がある。

3 提案手法

3.1 サブワードクエリ

図 1 に示すように、同一化合物をあらわす化合物名は部分的に共通要素をもつ傾向がある。そこで、検索クエリおよび検索元データに含まれるそれぞれの化合物名を幾つかの部分（サブワード）に分割し、分割結果を使って OR 検索をおこなう。この方法を本稿ではサブワードクエリと呼ぶ。

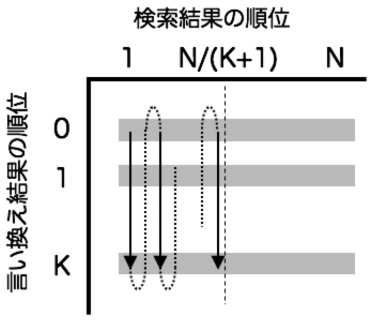

図 2: 検索結果の統合：$K+1$ 個の N-best 検索結果を 1 個の N-best 結果へ統合する。図中では便宜的に言い換え元クエリを 0 位としている。

サブワードへの分割の仕方によって検索結果が変化する。今回は、以下の 2 種類のサブワード分割手法を適用した。

- **SYMBOL**：クエリ内の記号文字の箇所で分割
- **SP**：SentencePiece[1] を使ったデータ尤度に基づく分割 [3]

3.2 言い換えによるクエリ拡張

図 1 の青色矢印の要素がサブワードクエリによって考慮されるが、赤色矢印の要素は無視される。この要素へ対応するために言い換え処理に基づくクエリ拡張をおこなう。具体的には、まず、元の検索クエリに対してサブワードクエリを作成する。つぎに、サブワードクエリに対して言い換え処理を実行することで新しいクエリを生成し、元クエリと新クエリのそれぞれを使って検索をおこなう。その後、各検索結果を統合することで 1 個の最終的な N-best の検索結果を得る。

言い換え結果が順位付きで得られ、K-best の新クエリが得られたとき、次の方法によって、元クエリと新クエリをあわせた $K+1$ 個の検索結果をひとつに統合する（図 2）。

- 検索結果の統合：

  検索結果の順位を優先して、$K+1$ 個の各クエリの検索結果を横断しながら、$N/(K+1)$ 位までの結果を統合することで 1 個の N-best の検索結果を得る。

[1] https://github.com/google/sentencepiece

4 評価実験

4.1 実験設定

4.1.1 データセット

PubChem[2] に含まれる化合物名 277,765,611 件を検索元データとし、このデータに含まれる化合物のうち、同一名称ではなく別称となる化合物名 1 万件を検索クエリとした。2 節でも述べたように、検索クエリが検索元データから見て別称となる状況を想定するため、検索元データには、検索クエリと同一名称ではないが検索クエリと同一の化合物をあらわす化合物名が 1 件以上含まれている。検索クエリあたりの、検索クエリと同一化合物をあらわす化合物名の平均登録数は 2.3677 である。

4.1.2 検索環境

検索処理には Elasticsearch[3] を用いた。具体的には Elasticsearch の match クエリを用いて全文検索として検索を行った。Elasticsearch ではデータベースのことをインデックスと呼ぶ。今回の実験では、SYMBOL、SP のそれぞれでサブワード分割した化合物名を各インデックスに登録し検索対象としている。

4.1.3 サブワード分割

SP のサブワード学習の学習データとして検索元データ全体を利用した。また学習時の語彙数の設定は 4,000 とし、unigram モデル [3] で学習を行った。

4.1.4 言い換え処理

元の検索クエリから言い換え結果を得るためにニューラル機械翻訳モデル fairseq[2, 1][4] を利用した。モデル学習の学習データは次のように準備した。すなわち、検索元データ内で同一化合物をあらわす化合物名を取り出し、サブワード分割処理の後、それらから擬似的な（言い換え元、言い換え後）対を作成した。これによって、検索元データから 447,124 件の言い換え対を作成し、学習に用いた。

[2] https://pubchem.ncbi.nlm.nih.gov/
[3] https://www.elastic.co/jp/
[4] https://github.com/facebookresearch/fairseq

表 1: 言い換え学習時のモデルパラメータ

epoch	100.0
batch size	64
clip norm	0.100
learning rate	0.25
dropout	0.200

fairseq の詳細設定は以下の通りである。翻訳モデルとして Gehring ら [2] の Convolutinal Neural Networks を利用した。設定パラメータを表 1 に示す。

上記モデルを適用することで、各クエリごとに言い換え結果として 5-best を獲得し、検索の性能評価に用いた。

4.2 実験結果

4.2.1 サブワードクエリの実験結果

表 2 にサブワードクエリの実験結果を示す。まず、N を大きくするに従って、Recall が向上することがわかる。また、サブワード分割手法を比較すると、SP が SYMBOL よりも明らかに Recall が高い。SYMBOL は記号箇所での分割を行う手法であるので、化合物名内の記号数に分割が左右される。それに対して SP は記号箇所以外でも分割を行うため、SYMBOL よりも様々サブワードが得られる。その結果、各サブワードが検索に有効な情報を保持した状態であるため良い結果を出したと考えられる。

MAP 指標で見ても SP が SYMBOL よりも良い結果となっている。ただし、Recall とはちがい、N を変化させても MAP 値の変動はわずかであった。

4.2.2 クエリ拡張の実験結果

言い換えによるクエリ拡張の実験結果を表 3 に示す。先述のように SYMBOL よりも SP の有効性が確認できるため、ここでの実験では SP のみで実施した。表 2 と表 3 の比較から、サブワードクエリにクエリ拡張を施すことで大幅に検索性能が向上することが確認できる。

特に、Recall に関しては、N=1,000 の時点で 0.9 を超えることが確認できる。この結果は、当初目的である同一性判定処理の候補絞り込みの観点から考えると、1,000 件を対象とした同一性判定処理を実行するだけで、9 割以上の事例を考慮することができることをあらわしている。

表 2: サブワードクエリの実験結果

	SYMBOL		SP	
N	Recall	MAP	Recall	MAP
1,000	0.3598	0.0352	0.6033	0.1562
2,000	0.4159	0.0352	0.6476	0.1563
3,000	0.4441	0.0353	0.6729	0.1563
4,000	0.4640	0.0353	0.6890	0.1563
5,000	0.4792	0.0353	0.7027	0.1563
6,000	0.4915	0.0353	0.7130	0.1563

表 3: 言い換えによるクエリ拡張の実験結果

	SP	
N	Recall	MAP
1,000	0.9309	0.4153
2,000	0.9396	0.4154
3,000	0.9446	0.4154
4,000	0.9474	0.4154
5,000	0.9495	0.4154
6,000	0.9516	0.4154

4.2.3 クエリ拡張の事例観察

クエリ拡張のために得られた言い換え事例を観察する。以降、空白はサブワード分割箇所をあらわし、「__」は元単語の先頭情報をあらわす。

表 4 に、検索クエリ「__-(2- chloranyl -5- oxidanylidene - furan -2- yl) ethanoi c __ acid」とその言い換え結果を示す。検索元データに登録されている、検索クエリと同一化合物をあらわす化合物名もあわせて示す。この例では、サブワードクエリ単体では、N=6,000 の検索結果内に別称 1 ～ 4（検索クエリと同一化合物をあらわす化合物名）をひとつも含めることができなかった。しかし、クエリ拡張によって、別称に近いクエリを獲得することに成功しており、特にこの例の場合、第 1 位の言い換え結果が別称 3 と完全に同一の文字列をとなっている等、検索すべき別称を高い検索順位で検索することに成功していた。

別の例を表 5 に示す。こちらの 2 件はどちらとも、クエリ拡張後でも N=6,000 の検索結果内に検索すべき別称をひとつも含めることができなかった例である。

表 4: クエリ拡張の例 1

検索クエリ	__ 2-(2- chloranyl -5- oxidanylidene - furan -2- yl) ethanoi c __ acid
別称 1	__ 2-(2- chloro -5- oxo furan -2- yl) acet ic __ acid
別称 2	__ 2-(2- chloro -5- oxo -2- furanyl) acet ic __ acid
別称 3	__ 2-(2- chloro -5- keto -2- furyl) acet ic __ acid
別称 4	__ 2-(2- chloro -5- oxo -2- furyl) acet ic __ acid
K-best	検索クエリの言い換え
1 位	__ 2-(2- chloro -5- keto -2- furyl) acet ic __ acid
2 位	__ 2-(2- chloro -5- oxo -2- furyl) acet ic __ acid
3 位	__ 2-(2- chloro -5- oxo -2- furanyl) acet ic __ acid
4 位	__ 2-(2- chloro -5- oxo - furan -2- yl) acet ic __ acid
5 位	__ 2-(2- chloro -5- oxo furan -2- yl) acet ic __ acid

表 5: クエリ拡張の例 2

検索クエリ	__ formi c __ acid	__ a l l an t o in
別称 1	__ methano ic __ acid	__ 1- [2,5- bis (oxidanylidene) imidazolidin -4- yl] urea
別称 2	なし	__ (2,5 - dioxo imidazolidin -4- yl) urea
別称 3	なし	__ (2,5 - dioxo -4- imidazolidinyl) urea
K-best	検索クエリの言い換え	
1 位	__ formi c __ acid	__ 2- amino enal
2 位	__ formi c __ acid ; form ic __ acid	__ 2- hydroxy enal
3 位	__ 2- form ic __ acid	__ 2- azanyl butanal
4 位	__ formi c __ acid ; m ethanoate	__ 2- azanyl pyridine -1- one
5 位	__ form ic __ acid	__ 2- amino benzaldehyde

例のように、化合物名が長くない場合にこのようになる事例が多く見られた。このような事例では、検索クエリと検索すべき別称間の名称の隔たりが大きく、言い換え処理をおこなってもこの差を吸収できていなかった。

5 おわりに

化合物の同一性判定のための判定候補の絞り込み手法として、サブワード分割と言い換えを利用した手法を提案した。評価実験から、提案手法によって N=1,000 の検索結果内に 9 割以上の別称化合物名を含めることができることを確認した。今後は、言い換え処理でも検索できなかった事例の追加、および、絞り込み結果を用いた同一性判定実験をおこなっていく計画である。

参考文献

[1] Jonas Gehring, Michael Auli, David Grangier, and Yann N Dauphin. A Convolutional Encoder Model for Neural Machine Translation. *ArXiv e-prints*, 2016.

[2] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional Sequence to Sequence Learning. *ArXiv e-prints*, 2017.

[3] 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 Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, 2018.