自動翻訳を使った英語論文の書き方 v3

塩谷 亮太

チェックシート

- 工事中

はじめに

はじめに

- この資料では自動翻訳などのツールを使って英語論文を書く方法を説明します

- 主に以下のツールを使うことを想定しています
  - 自動翻訳
    - DeepL
    - Google 翻訳
  - 文法チェッカー
    - Grammarly（汎用の英文法チェッカー）
    - Writeful（overleaf 用の英文法チェッカー拡張）
  - Google 検索

モチベーション
なぜ翻訳ベースで書くのか？（１）：速いし楽だから

- 後述するように，作業の大半は「良い日本語の論文を書く」こと
  - ◇ 「良い論文」は日本語でも英語でもほぼ同じ構造を持つ
- これが済むとかなりの短時間で機械的に英語にできる
  - ◇ １０ページ程度の論文なら数日程度
- 一方で，「良い日本語の論文を書く」には１月～２月はかかる
  - ◇ 何度も何度も書き直しを経る必要がある
  - ◇ このサイクルは母国語でやった方が圧倒的に速い

モチベーション
なぜ翻訳ベースで書くのか（２）：質が高いから

1. 英語では文章を直す際のサイクルがどうしても長くなる
   - ◇ 非母国語での思考は，速度がどうしても落ちる

2. 英語でうまく書けない場合，そもそも無かったことにされがち
   - ◇ 目的の表現をどう書いたら良いかわからない場合に，そもそも文章に書かない人がとても多い

3. 自動翻訳のレベル
   - ◇ 平均的な東大生の英作文能力よりも，「適切に書かれた日本語を入力とした場合の」自動翻訳の方が，もうレベルが高い
   - ◇ 日本人がやりがちな誤った書き方や不自然な表現が現れにくい

実際の作業の流れ

- 主な流れ：
  1. 日本語で論文を書く（これは別途終えているものとする）
  2. 日本語の論文を英語向けの日本語に書き換える
  3. 自動翻訳を使いながら英語にする
  4. 文法チェッカーを使って修正をする
  5. （英文校正に出す）

- 2. の日本語から日本語への書き換えが，3. の英語にする部分よりも作業量が多い

- 繰り返しているうちに，最初から英語でも書けるようにもなってくる
  - ◇ でも学生さんは，いきなり英語で書くのはやめておいた方がよいと思う

注意

- この資料はあくまで「英語論文の質を上げる」ことが目的
- 著者の英語の能力を上げる事を目的としていない
  - ◇ 純粋に本人の英語能力を上げることが目的であれば,
    最初から英語でずっと書いている方が良いかもしれない

もくじ

英語にできる日本語に書き換える

日本語から英語になる日本語への書き換え

- 良く書けている論文は言語によらず同様の構造を持つ
  1. 各文は短く簡潔である
  2. 各文は適切に接続されている
  3. 各文に主語や動詞，述語が明確にある

- 上記が満たされている場合，かなり機械的に日本語から英語に変換できる
  - ◇ まずは上記の条件を満たす日本語原稿を作る事を目指す
  - ◇ （実は，良い日本語の論文を書くこととあまり変わらない

- 以降では，上記の構造を実現する方法を順に説明

書き換えのポイント

1. 各文を短く簡潔にする
2. 各文は適切に接続する
3. 各文の主語や動詞，述語を明確にする
4. 英語になることを意識した日本語を書く

1. 各文を短く簡潔にする

- **短い文は全てを解決する**
  - ◇ ある意味この資料で一番大事なことはこれ
- 短く簡潔な文は,
  1. 誰が読んでも明確に意味が理解できる
  2. そして機械的にも翻訳できる

短い文にする

- なるべく各文は単文にする
  - ◇ 単文：主語と述語の組が１つだけの文
  - ◇ ２文の複文までは良いが，基本的に３文以上の複文は禁止
- 複文で「～が，」で文を繋ぐのは禁止
  - ◇ 逆接の意味で後ろに「しかし」がついているなら良い
  - ◇ そうではなくなんとなく繋がっているだけのことがある

書き換えのポイント

1. 各文を短く簡潔にする
2. 各文を適切に接続する
3. 各文の主語や動詞，述語を明確にする
4. 英語になることを意識した日本語を書く

文同士を接続する

- なにも考えずに文を短くするとぶつ切れになる
  - ◇ 元々なんとなく繋がっていた気がしていただけで，実は論理的に接続されていないとこうなりがち
- 文同士をきちんと接続する方法
  1. 接続詞を適切に入れる
     - □ 「したがって」「なぜなら」「しかし」
  2. 文内の論理的な繋がりを使う

論理的な繋がりを使った文の接続 1

- そこより前の文に出てくる単語や事象を入れると自然に繋がる
  - そこまでの文の内容に新しい情報を付け足す形にする
- 「A は B である．なぜなら B は C だからだ」の後に繋げる文を考える
  - 良い例：
    - 「この B は～という性質をもつ」
    - 「この C は一般に D である」
  - だめな例：
    - 「E は F である」（E も F も初登場なので繋がってない）

論理的な繋がりを使った文の接続２

- 各文内の前の方に，（なるべく近くの）既出の単語や事象を置く
  - ◇ 既出の単語が，長い文の後半で初めて出てくるのは良くない
  - ◇ その文を最後まで読まないと，接続関係がわからない

- また「A は B である．なぜなら B は **C** だからだ」の後に繋げる文を考える
  - ◇ だめな例：
    - □ 「～は～であり，そのため～は **C** である」
    - □ C が最後に出てくるので，そこまで読まないと関係がわからない
  - ◇ 良い例：
    - □ 「～は **C** である．なぜなら～は～であるためである」

論理的な繋がりを使った文の接続３

- 最初に要素や単語を列挙してから，ぶらさげる
  - この後ろに列挙した要素の説明がくることが自然に伝わる
- 例：
  - 「A には B と C がある」
    - 「B は～である」
    - 「一方で C は～である」
  - 「～は以下の手順で行われる」
    - 「1.」「2.」「3.」
  - 「～は以下の２つの理由からなる」
    - 「1.」「2.」

書き換えのポイント

1. 各文を短く簡潔にする
2. 各文を適切に接続する
3. 各文の主語や動詞，述語を明確にする
4. 英語になることを意識した日本語を書く

各文の主語や動詞，述語を明確にする

- 日本語は主語や述語（目的語）を省略して書けてしまう
  - ◇ 日本語論文であっても，本来そのような曖昧さは排除すべき
  - ◇ 多少冗長であっても，意味に紛れがないよう略さずにきちんと書くべきである
- 主語や述語がない日本語は適切に英語に翻訳できない
  - ◇ 機械翻訳はもちろんだが，人間にも難しい
- まず各文を確認して，主語や述語が欠けていないかを確認
  - ◇ 必要に応じて追加する
  - ◇ 自明な場合でも「それ」などを使って明示することで英語になりやすくはなる

書き換えのポイント

1. 各文を短く簡潔にする
2. 各文を適切に接続する
3. 各文の主語や動詞，述語を明確にする
4. 英語になることを意識した日本語を書く

英語を意識した日本語

- 英語になることを見越して日本語を書くことで，より自然になる場合もある
  - 基本的には文を短くするだけで解決可能
- 例：
  - 関係代名詞による修飾を意識する
    - 少し長い修飾節でも，英語になった際の関係代名詞次第ではトップヘビーな文を避けられる
    - 後置修飾の形を想像しながら書く
  - 無生物主語を積極的に使う
    - 日本語では不自然だが，英語では普通
    - 提案手法の名前や proposed method を主語にするなど

主題を表す「は」を書き換える

- 日本語の「は」には，主語と主題の２通りの使い方がある
  - ◇ たとえば「象は鼻が長い」の場合，
    - □ 主題：象は
    - □ 主語：鼻が
- この「主題」の提示に１：１にきれいに対応する英文法がない
  - ◇ 翻訳が不自然になりがち
- 主題から主語に書き換える自然になることが多い
  - ◇ 「象は鼻が長い」→「As for elephants, their noses are long」
  - ◇ 「象は長い鼻をもつ」→「Elephants have long noses」

主語が大きいトップヘビーな文を避ける

- 元の日本語を書き換えたり，DeepL の候補の選択などでなんとかする

- 元：
  メモリ順序管理についても、大半の命令は順序違反を起こさないことを利用した軽量化手法やコンパイラの解析に基づく軽量化手法が提案されている。
  For memory order management, lightweight methods that take advantage of the fact that most instructions do not cause order violations and lightweight methods based on compiler analysis have also been proposed.

  主語を含む名詞節がとてつもなく長い

- 修正後：
  大半の命令は順序違反を起こさないことを利用した軽量化手法やコンパイラの解析に基づく軽量化手法などを含む、メモリ順序管理についての既存手法が存在する。
  There are existing methods for memory order management, including lightweight methods that take advantage of the fact that most instructions do not violate order, and lightweight methods based on compiler analysis

  「There are existing methods」として methods 後ろに持って行くことで，後置修飾できる形にする

自動翻訳を使いながら英語にする

手順

- パラグラフ単位で日本語から英語にしていく
- 元の日本語はコメントアウトしておくとよい
  - 大抵のエディタや overleaf では範囲指定して「ctrl+/」でコメントの切り替えができる

ながれ

1. DeepL に１パラグラフ分の日本語をいれる
2. 大抵そのままでは使えない英語が出てくる（後述）ので，日本語を直す
3. 英語を手直しする
4. 出来上がった英語を Google 翻訳にかけて日本語に戻す
   - ◇ 意図した意味になっているか確認する

逆翻訳時になぜ違う翻訳を使うのか
（2022年の場合であり，今後変わる可能性も高い）

- 異なる翻訳エンジンを逆翻訳時に使うことで精度を上げる
  - ◇ 同じ翻訳エンジンを使うと，逆翻訳では単に元の文が復元される確率が高い
  - ◇ 同一のネットワークを使って翻訳しているからかもしれない
- 最初の翻訳は DeepL を使う
  - ◇ 現状 DeepL と Google 翻訳では前者の方が質が高い
- 逆翻訳は Google 翻訳を使う
  - ◇ DeepL は入力が適当な日本語でも補完して，文法的には正しいが意味的におかしい文を生成しがち
  - ◇ Google 翻訳は入力が適当だと翻訳をミスしてくれる

出てきた英語の確認と訂正１

- 大抵そのままでは使えない英語が出てくる
  - 意味がおかしい
  - 英語として不自然
  - 文がたまに欠落する
- 基本的には,
  - 元の日本語を１つずつ書き換える
    - この過程で長すぎる文を短くして単純化したり，主語の欠落に気づいて補ったりすることも多い
  - どうしようもないものは英語を直接書く

出てきた英語の確認と訂正２

- 他にも以下を確認する
  1. 冠詞
  2. 名詞的な動詞の使用
  3. 一般的でない単語/文法の使用
  4. 「各文を適切に接続する」で説明した接続の確認
- 以降でそれぞれを説明

冠詞

- 冠詞の判断は（この資料の使い方の限りは）自動翻訳には不可能
  - ◇ 定冠詞/不定冠詞/複数形無冠詞のどれを使うか？
- その名詞が文脈上で読者から明らかどうかは，わからないから
  - ◇ そこまでの論文全体の内容から判断する必要がある
  - ◇ パラグラフ単位の入力のみでは原理的に判断不能
    - □ 日本語には冠詞がないので，この情報が大概欠落している
- したがって，冠詞が適切かどうかは全ての文において手動で確認する必要がある

名詞的な動詞の使用

- 「～を行う」から発生しがち
  - ◇ たいてい「～する」に変更すれば自然と動詞になる
- 例：
  - ◇ このプログラムはその値の大きさの判断を行う →
    This program makes a determination of the magnitude of the value.
  - ◇ このプログラムはその値の大きさを判断する →
    This program determines the magnitude of the value.

一般的でない単語/文法の使用

- 見たこともない単語や用法が出てきた場合，なにかおかしい可能性が高い
  - うまく訳せなかったための翻訳エンジンの悲鳴な可能性がある
  - 文自体の複雑さや使用している元の単語を見直した方がよい
  - 基本的には日本語の時点で短く簡潔に書けば，ほとんどこれは起きない
- Google 検索にかけて確認
  - 日本人が書いたものが多くヒットした場合，日本人がやりがちな不自然な表現の可能性もある
- 論文は基本的に中学で習う程度の英語で書けるし，そうすべき
  - 専門用語以外は，可能な限り平易で簡潔な表現を用いる

「各文を適切に接続する」で説明した接続の確認

- 接続をよくするために，文の頭の方に既出の単語を持ってくる
- 方針：
  - 日本語自体を書き換えてなんとかする
    - 短い文であれば，比較的語順が維持される
  - DeepL の単語選択機能を使う
    - 翻訳後の単語をクリックすると，別の候補が出てくる
    - 候補をクリックすると，後続の翻訳が対応して書き換わる
    - これを使って接続の良い候補を選ぶ

文法チェッカーによる確認

文法チェッカーによる確認

- 手直しの際に文法の誤りが入ることは多いため，確認は必須
  - 下記のツールを使うと良い
    - Grammaly
    - Writeful
  - Overleaf を使っているなら，どちらのツールも自動で対応している
- ただし，指摘してくる事項は機械的に適用してはいけない
  - 特に冠詞など，文脈から判断されるものは指摘が間違っている事も多い

スペルチェックについて

- スペルチェック付きのエディタを使うこと
  - vscode ならアドオンを入れる
  - overleaf なら最初からついている
- 赤線が引かれたら無視しない
  - 固有名詞で正しい場合は全部辞書に登録する

英文校正について

英文校正について

- 出来上がった英文は，出来れば英文校正にかけた方が良い
  - ◇ 文法的に正しくても，不自然な言い回しになっている事などがある
- 業者には出来るだけ早めに，ゆっくりめのプランで出した方がよい
  - ◇ 超特急などでお願いすると校正の質が露骨に落ちることが多い
  - ◇ むこうも人間だし，１日で１０ページやれとかは無理がある

Word への変換

- 校正業者は PDF を直接校正できない場合が多い
  - ◇ 可能な場合でも，校正結果が見にくい事が多い
- PDF を Word に変換して提出すると良い
  - ◇ Word は PDF を変換して読み込むことができる
    - □ 変換精度は結構高い
  - ◇ Word の校閲機能を使って校正をしてもらう
  - ◇ Word の場合，書き換えにより文書がずれても割と平気

校正時のフォーマット

- ドキュメントクラスを差し替えて，シングルカラムに変更した方がよい
  - PDF から Word に変換した際にダブルカラムよりも形が崩れにくい
  - 校正者による書き換えでも形が崩れにくい

- 以下のようなマクロを使用して切り替えると良い

```
% 校正業者用にシングルカラム化
% 投稿時は ¥PROOFtrue をコメントアウト
¥newif¥ifPROOF
% ¥PROOFtrue

¥ifPROOF
    % 校正用シングルカラム化
    ¥documentclass[dvipdfmx,conference,onecolumn,12pt]{IEEEtran}
¥else
    % 元のドキュメントクラス
    ¥documentclass[dvipdfmx,conference]{IEEEtran}
¥fi
```