コーパスに基づく機械翻訳に関する研究

土居, 誉生

(Degree)
博士（工学）

(Date of Degree)
2007-03-25

(Date of Publication)
2012-04-09

(Resource Type)
doctoral thesis

(Report Number)
甲3965

(URL)
https://hdl.handle.net/20.500.14094/D1003965

博士論文

コーパスに基づく機械翻訳に関する研究

平成 19 年 1 月

神戸大学大学院自然科学研究科

土居 誉生

概要

近年，対訳コーパスから自動的に翻訳システムを構築するコーパスベースの機械翻訳技術の研究開発が盛んになってきており，実用化に向けた一層の性能向上が期待されている．一般に機械翻訳システムの性能向上のためには，翻訳エンジン，翻訳前処理，翻訳後処理の 3 側面での工夫による効果が期待できる．本論文では，グラフベースの用例検索手法を使った翻訳エンジンの話題を中心に，コーパスベース翻訳エンジン共通の課題への前・後処理による対策を取り上げ，コーパスベース翻訳の性能向上について論じる．

用例翻訳や統計翻訳といったコーパスベース翻訳では，基本的に単語や句などの小さな単位の翻訳知識を獲得しそれを使って翻訳を実行する．一方，翻訳すべき入力文とほとんど同じ文とその訳文のペアがコーパス中に存在すれば，そのペアを直接利用することで非常に良い訳が得られる．この原理に基づく文単位の用例翻訳を想定し，その実現に必要な課題に取り組む．用例翻訳では，入力表現に最も類似した対訳用例を抽出し，その翻訳表現の一部を変更して入力に対する訳を生成する．通常，対訳表現の単位は句である．句の組合せにより多くの入力文に対応可能であるが，各部分の誤りの合成や組合せの不整合により，不適切で不自然な訳文が生成されてしまう危険がある．一方，用例単位として文を使う場合，その危険性を抑えることができる．入力文全体に類似した文単位の用例が見つかれば，正確で自然な翻訳文を得ることができる．もちろん文は句に比べ，より長い単位であり汎用性が劣るため，文単位の用例翻訳はカバレッジの面で短所を持つ．文単位の用例を使って十分な翻訳カバレッジを得るためには，大規模な対訳コーパスを用意しなければならず，大規模コーパスから用例を検索するための効率的な検索手法が必須となる (第 2 章)．

ここでは編集距離に基づく類似度を用いた文単位の用例翻訳システムを想定し，そのシステムが大規模コーパスを扱うための効率的な検索手法を提案する．対象となる検索処理は，コーパス中の原言語文を候補文とし，入力文との距離が閾値以内で最小の候補文を全て求めることである．その目的のために全ての候補文について逐次的に距離を計算するのでは時間がかかり過ぎてしまう．そこで候補文集合の分割，単語グラフ，A*アルゴリズムを利用した効率的な検索手法を提案する．提案手法では 2 文間の逐次的な照合は行わず，グラフ化された複数の候補文と入力文との照合を同時並行的に進める．この手法では編集距離の定義と与えられた閾値に関して検索もれはな

い．提案検索手法では，内容語数と機能語数を基に候補文をグループ分けする．これにより入力文の内容語数と機能語数および距離閾値を基にしたグループ単位での枝刈りが可能となる．各グループ毎に複数の候補文が一つの単語グラフにまとめられる．単語グラフは有向グラフであり，先頭ノードから最終ノードに至る可能な道筋と候補文が互いに対応する．複数の文に共通な単語列がグラフ中で一つにまとめられ，ノード数が最小となるように圧縮される．グループ内の検索は，単語グラフの先頭ノードから最終ノードまでの可能な全経路について，各経路に現れる単語列と入力単語列との照合結果の中から編集距離を最小にするものを探索することである．この探索問題の解法に A*アルゴリズムを用いる．一般に A*アルゴリズムでは，問題状態集合の中から最終コストの下限の推定値が最小のものが選ばれ継続状態に展開される．ここで対象とする問題では，状態は，単語グラフの経路と入力文との照合の途中経過を意味する．コスト推定の際，単語グラフ内では全ての候補文の単語数が等しいという条件が利用される (第 3 章).

旅行会話に関する数十万文規模の対訳コーパスを使った日英翻訳実験を通して，提案検索手法でもって実装した翻訳システムの性能を評価した．利用したコーパスは，海外旅行者向けのフレーズブックに相当する内容の日本語文とその英訳からなる．翻訳結果の品質評価のために客観評価スコアと主観評価ランクを用いた．また処理効率の評価のために，Pentium4/2GHz の通常のパーソナル・コンピュータ上での翻訳処理時間を計測した．15 万対訳からなる学習セットを使って基本的な翻訳性能を評価した結果，最高ランクに分類される訳文の割合が 71% と大きく，高い翻訳品質が示された．翻訳処理時間は，平均 0.2 秒，最大 3.3 秒であり，提案した用例検索手法により効率的な処理が実現された．コーパスサイズと翻訳性能の関係を調べるために，学習セットの大きさを 1.9 万から 30 万対訳の間で変えて性能評価を行った．結果として各指標は，コーパスサイズが大きくなるほど大きなカバレッジと高い翻訳品質が得られることを示した．一方，処理時間はコーパスサイズのほぼ 1/2 乗のオーダーに抑えられている．編集距離基準を用いた文単位の用例翻訳システムは，大規模コーパスを使うことにより高品質の翻訳能力を持ち，提案検索手法を使うことにより効率的な翻訳処理が可能となっている (第 4 章).

一般に機械翻訳システムは入力文が長くなると誤りが多くなる．第 4 章の実験でもその傾向が確認されている．しかし入力文が長くて翻訳誤りが起こる場合でも，分割した各部分に対しては翻訳が成功する可能性がある．入力文を互いに独立性の高い部分に分割できれば，それぞれの翻訳結果を同じ順番に並べることにより入力全体を

翻訳することができる．そこで，機械翻訳システムを使ったより良い翻訳が可能となるように，前処理で入力文を分割することにより翻訳システムに渡す単位を適正化するアプローチをとる．文分割に関する従来の研究では，分割点周辺の N グラムなどの単語の連接の特徴に基づいた手法が多い．それに対して提案手法では，単語連接に基づく分割を補うための別指標として類似度を導入する．当分割手法では，N グラムに基づき分割点の候補を生成し，コーパスを使って計算した類似度に基づき分割点の最良の組合せを選択する．この選択は，コーパスベース翻訳システムは，学習コーパスに存在する文と類似した文は正しく翻訳することができるという仮定に基づいている．当手法の評価のため，句単位の用例翻訳，文単位の用例翻訳，統計翻訳の異なる手法による 3 つのコーパスベース翻訳システムによる英日翻訳実験を行った．実験結果は，いずれのシステムに対しても提案分割手法が有効であり，類似度の使用は翻訳品質を改善することを示している (第 5 章)．

コーパスベース翻訳では，特徴的な翻訳誤りとして，入力文と関連のない語が訳文に湧き出す問題が観察される．第 4 章の実験結果からも翻訳誤りの多くがこのタイプの誤りに分類される．この問題は単語アラインメントというコーパスベース翻訳に本質的な処理に由来する．この湧き出し誤りへの対策として，後編集により自動修正するアプローチを提案する．後編集アプローチには，特定の翻訳システムに限らず適用可能という利点がある．提案手法は誤り語を自動で削除する．この手法では，まず異なる言語の単語間の対応確率を示す単語翻訳モデルを利用し，出現の期待値が閾値以下の訳語を誤り語候補として検出する．次に対訳コーパスから得られた用例を利用して誤り語候補の正しさを検証する．つまり誤り語候補が特定の類似用例に存在し，その用例において入力文と共通する語句から相対的に高い期待値が得られれば，その誤り語候補は誤りではなく正しい訳語と判断される．最終的に残った誤り語が訳文から削除される．日英および中英翻訳を対象とした複数のシステムを使った実験で，誤り語自動削除による翻訳精度向上効果が確認された (第 6 章)．

以上のように，本論文ではコーパスベース翻訳の性能向上のための手法について論じた．単語グラフを使った用例検索手法により高品質で高速な翻訳システムを実現することができた．さらに関連する翻訳誤りに対して，前処理による入力文分割，後処理による湧き出し語削除の対応策を提案し，複数の翻訳システムにおいて効果を確認した．

目次

第 1 章　はじめに 1
1.1　機械翻訳 1
1.1.1　ルールベース翻訳 1
1.1.2　コーパスベース翻訳 2
1.2　研究の目的とアプローチ 4
1.3　本論文の構成 4

第 2 章　文単位の用例翻訳 6
2.1　文単位の用例翻訳の意義と課題 6
2.2　編集距離を使った用例翻訳 7
2.2.1　構成 7
2.2.2　翻訳処理 8

第 3 章　グラフベースの検索手法 12
3.1　用例検索 12
3.1.1　候補文集合の分割 12
3.1.2　単語グラフ 13
3.1.3　A*アルゴリズム 13
3.2　探索 14
3.2.1　状態空間表現 14
3.2.2　探索アルゴリズム 17
3.2.3　実行例 19

第 4 章　グラフベースの検索手法を使った翻訳システムの評価 21

4.1 実験条件 21
4.2 基本的な翻訳性能 22
4.3 人間の英語能力との比較 23
4.4 他の文単位の用例翻訳との比較 24
4.5 翻訳例と誤り分析 26
4.6 翻訳カバレッジ 28
4.7 大規模コーパスを使うことによる効果 29
4.8 DP マッチとの比較 31
4.9 まとめ 32

第 5 章 前処理による翻訳単位の適正化 34
5.1 入力分割による翻訳精度向上の可能性 34
5.2 N グラム言語モデルと類似度を用いた分割手法 36
5.2.1 分割文表現 36
5.2.2 構成 36
5.2.3 N グラム言語モデルに基づく指標 37
5.2.4 類似度 38
5.2.5 分割候補の生成 40
5.2.6 最良分割の選択 42
5.3 評価 43
5.3.1 実験条件 43
5.3.2 入力分割による翻訳品質向上効果 46
5.3.3 類似度を用いた選択の効果 47
5.3.4 類似度を用いた選択のコスト 48
5.3.5 シソーラスを用いることによる効果 49
5.3.6 まとめ 49

第 6 章 後処理による翻訳誤り対策 55
6.1 コーパスベース翻訳における特徴的な誤り 55
6.2 湧き出し語問題 56
6.2.1 用例翻訳における問題 57
6.2.2 統計翻訳における問題 57

6.3 問題へのアプローチ 58
6.3.1 後編集による修正 58
6.3.2 関連研究 58
6.4 単語翻訳モデルを用いた訳語削除手法 60
6.4.1 単語翻訳モデル 60
6.4.2 単語翻訳モデルによる削除語候補検出 61
6.4.3 対訳用例による削除語の制限 61
6.4.4 削除の実施 63
6.5 評価 63
6.5.1 実験条件 63
6.5.2 翻訳品質への効果 64
6.5.3 翻訳妥当性との関連 65
6.5.4 統計翻訳への効果 67
6.5.5 削除例 68
6.5.6 まとめ 69

第 7 章 おわりに 76

謝辞 79

参考文献 80

本論文に関する原著論文 85

第 1 章

はじめに

近年，対訳コーパスから自動的に翻訳システムを構築するコーパスベースの機械翻訳技術の研究開発が盛んになってきており，実用化に向けた一層の性能向上が期待されている．対訳コーパスとは，2 つの言語の文のペアを集めたデータベースの一種を指し，各ペア中の 2 つの文は互いに対訳関係にある．コーパスベース翻訳では，対訳コーパス中の各ペアを翻訳の模範として，翻訳処理に必要な知識を自動的に獲得する．本論文はコーパスベース翻訳の分野における翻訳性能向上のための手法について論じる．

1.1 機械翻訳

コーパスベース翻訳以外の代表的な機械翻訳手法はルールベース翻訳と呼ばれる手法である．またコーパスベース翻訳の代表的な手法には用例翻訳と統計翻訳がある．各機械翻訳手法について概観する．

1.1.1 ルールベース翻訳

ルールベース翻訳は，機械翻訳システムの歴史の中で早くから用いられた手法であり，現在でも商用翻訳アプリケーションの主流として利用されている．ルールベース翻訳では，翻訳の規則を人手により作成し，その規則を使って翻訳を実行する．ルールベースのアプローチは，翻訳知識を人手で構築するという面から，コーパスベースのアプローチとは対極にある．例えば日英機械翻訳システム ALT-J/E [21] では，日

本語格フレームに英語表現を対応させる約 1 万 5 千件のパターンを人手で記述している．ルールベース翻訳には，人手で翻訳規則を作成しなければならないがゆえに，保守性と移植性を含めた開発効率に欠点がある．

- 人手で翻訳知識を作成するコストがかかる．一般に機械翻訳の対象となる自然言語には，少数の規則では対応しきれない例外的な現象が多く，それらに対応して良い翻訳を出そうとすると大量の規則が必要となってしまう．規則が増えると，1 つの規則の追加や変更が与える影響をとらえることが難しくなり，開発および保守にかかるコストが指数関数的に大きくなる．
- 翻訳規則を簡単に移植することができない．あるドメインである原言語からある目的言語への翻訳システムが出来上がったとしても，そのシステムを他のドメインや他の言語に対応させるには，人手で規則を書き換えるか作り直す必要がある．大量の規則からなるシステムの移植には大きなコストがかかる．

1.1.2　コーパスベース翻訳

コーパスベース翻訳では，翻訳知識の獲得に人手が不要であるため，ルールベース翻訳で見られた欠点が克服される．つまりコーパスベース翻訳は，保守性と移植性を含めた開発効率に優れる．翻訳システムを構築するには対訳コーパスを用意しさえすればよい．目的とするドメインの目的とするとする 2 つの言語の対訳コーパスを用意すれば，そのドメインのその 2 言語間双方向の翻訳システムが実現できる．対訳コーパスを構成する対訳は，特定の機械翻訳システムで使われる翻訳規則に比べ，人間社会でより広く使われる一般的な知識表現であり，より容易に集めることができる．人手で新たに作成する場合でも，対訳文を作るには 2 言語の理解が必要だが，翻訳規則を作るには 2 言語の理解に加え，対象となる翻訳システムに固有の知識が必要である．さらに翻訳規則の追加修正には他の規則との関係を注意深く考慮する必要があるが，対訳文の追加修正の際には他の対訳との関係を考慮する必要はない．

コーパスベース翻訳では，大量のデータを使った機械学習など大きな計算機パワーを要する処理が多い．近年の計算機の高性能化によりコーパスベース翻訳が現実的なものとなり，その研究が盛んになってきた．また対訳コーパスが充実してきたこともコーパスベース翻訳の研究を進める要因になっている．逆にコーパスベース翻訳の研究によって対訳コーパスのさらなる整備も進んでいる．

コーパスベース翻訳は大きく用例翻訳と統計翻訳に分けられる．

用例翻訳

用例翻訳の研究は，アナロジーに基づく機械翻訳の概念の提唱 [28] に始まる．用例翻訳では，原言語の表現とそれに対応する目的言語の表現からなる対訳を使って翻訳を行う．この対訳表現のことを用例と呼ぶ．用例は対訳コーパスから自動的に抽出される．用例翻訳では，翻訳処理に先立ち，あらかじめ用例のデータベースを作成しておく．翻訳処理は大きく 2 つのフェーズに分けられる [39]．まず入力表現に最も類似した原言語表現を持つ用例を検索する．次に，用例の原言語表現と入力表現の差異に応じて，用例の目的言語表現を書き換えることにより翻訳結果を得る．用例翻訳では類似度の判定に際してシソーラスを利用することも特徴として挙げられる．

また対訳コーパスで与えられた文のペアがそのまま用例として利用されることはほとんどなく，文のペアから，分解や抽象化の過程を経て用例が取り出される．通常この用例の単位は句であり，句の訳を組み合わせて入力文の訳を作ることになる．

統計翻訳

統計翻訳では暗号解読の考え方により翻訳を行う [8]．つまり観察された記号列を復号することにより目的の記号列を得る．ここで観察された記号列は原言語の入力文であり，目的の記号列が目的言語の翻訳文である．入力文 F に対して求める訳文 E は，次の条件付き確率 $P(E|F)$ を最大化した式で表される．

$$\underset{E}{argmax}\, P(E|F) = \underset{E}{argmax}\, P(E)P(F|E) \tag{1.1}$$

右辺の $P(E)$，$P(F|E)$ で示す確率は，それぞれ言語モデル，翻訳モデルと呼ばれるモデルにより与えられる．言語モデルは与えられた記号列が目的言語の文となっている確率を示す．一方翻訳モデルは，ある目的言語の文が与えられたときに，その翻訳としてある原言語の文が生成される確率を示す．言語モデルは目的言語コーパスから，翻訳モデルは対訳コーパスから自動的に推定される．モデルの種類に応じて様々な機械学習手法が使われる．

これらのモデルは基本的に単語などの小さな単位を扱う．つまり，ある単語がある単語に訳される確率，ある単語とある単語が連接する確率などを積み重ねて文としての確率を計算する．最近では単位として句を用いることが多くなっている [50, 30]

が，その単位は，用例翻訳で使われる用例の単位よりも小さいことが多い．

1.2　研究の目的とアプローチ

用例翻訳や統計翻訳といったコーパスベース翻訳では，基本的に単語や句などの小さな単位の翻訳知識を獲得しそれを使って翻訳を実行する．一方，翻訳すべき入力文とほとんど同じ文とその訳文のペアがコーパス中に存在すれば，そのペアを直接利用することで非常に良い訳が得られる．この原理に基づく文単位の用例を使う翻訳方式を想定し，その実現に必要な課題に取り組む．我々はこのアプローチの主たる課題は用例検索にあるととらえ，翻訳品質の面から効果的な検索指標とその指標に従った効率的な検索手法を考える．

用例検索は，用例翻訳システムにおいて中核となる翻訳エンジン部に位置づけられる．一般に機械翻訳システムの性能向上のためには，翻訳エンジンのみならず，翻訳前処理によって入力を翻訳しやすい形に変換する手法，翻訳後処理によって結果をより適切な訳に変換する手法による効果が期待できる．本論文では，翻訳エンジンにおける用例検索，前処理におけるコーパスに基づく翻訳単位の適正化，後処理におけるコーパス翻訳に特徴的な翻訳誤りへの対策の各課題に取り組み，コーパスベース翻訳の性能向上について論じる．各課題は翻訳システム内の位置付けに沿って次のように並べられる．

- 翻訳前処理: 翻訳単位の適正化
- 翻訳エンジン: 効果的で効率的な用例検索
- 翻訳後処理: 翻訳誤りの修正

1.3　本論文の構成

本論文は次のように構成する．

第 2 章では，コーパスベース翻訳の中で文単位の用例翻訳の意義と課題について述べ，効率的な検索手法の必要性を明らかにする．また想定する文単位の用例翻訳システムを具体的に説明する．

第 3 章では，文単位の用例翻訳システムを実現するための，グラフベースの検索手法を提案しその構成とアルゴリズムを詳述する．

第 4 章では，グラフベースの検索手法でもって実装した用例翻訳システムを実験を通して評価する．ここでは文単位の用例翻訳の長所の確認とともに，その主たる課題が提案検索手法により克服されていることを確認する．またコーパスベース翻訳におけるいくつかの問題を観察する．

第 5 章では，長い入力文に対する翻訳品質を上げるために翻訳前処理による手法を提案し，複数のコーパスベース翻訳システムを使って評価する．

第 6 章では，コーパスベース翻訳における典型的な翻訳誤りに対処するために翻訳後処理による手法を提案し，複数のコーパスベース翻訳システムを使って評価する．

第 7 章では，以上についての成果のまとめを行う．

第 2 章

文単位の用例翻訳

2.1 文単位の用例翻訳の意義と課題

アナロジーに基づく機械翻訳 [28] の概念が提唱されて以来，このアイデアを具体化した用例翻訳方式が数多く提案されてきた [10, 22, 36, 42, 46, 48]．用例翻訳では入力表現に最も類似した対訳用例を抽出し，その対訳用例を入力に合わせて書き換えることにより翻訳結果を得る．通常この対訳用例の単位は句であり，句の訳を組み合わせて入力文の訳を作る．句はその汎用性から用例単位として適当であると考えられる．つまり句を組み合わせることにより多様な文に対応することができる．しかしながら入力文を句に分解し各部分の訳を組み合わせて翻訳文を生成する際，各部分の誤りや不整合など様々な誤りが混じり込む危険性が高くなる [39]．この危険性は，対訳コーパスに与えられた文のペアを対訳用例として使えば，回避することができる．以降，対訳コーパスで与えられた文のペアを分解・抽象化せずそのまま用例として使う場合，この用例を文単位の用例と呼ぶことにする．もし入力文に十分に類似した文単位の用例があれば，翻訳結果はコーパスで与えられた訳文に最小限の変更を施した文となる．この結果は正確で自然な表現となることが期待できる．もちろん文単位の用例を使うと句の利点である汎用性が失われ，低カバレッジの問題，つまり翻訳可能な入力文の範囲が限られてしまう問題が発生する．文単位の用例翻訳により十分な翻訳カバレッジを得るためには，大規模な対訳コーパスを使わなければならず，大規模コーパスから類似用例を抽出するために効率的な検索手法が必要となる．

本論文では編集距離に基づく類似度を使った文単位の用例翻訳システムのための検索手法を提案する．用例検索処理の効率的な実装は翻訳メモリと共通する課題であ

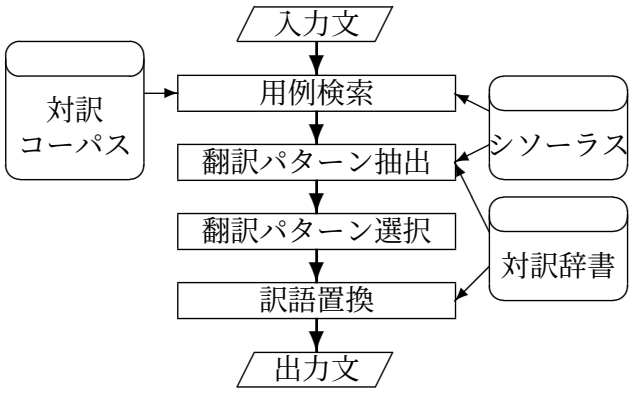

図 2.1. D^3 の構成

る．翻訳メモリに関する研究 [3, 13, 33, 35] では，1) 検索対象の絞り込み，2) 2 文間の照合アルゴリズム，の一方もしくは両方を論じている．1) を行なう場合，単語ベクタを使ったクラスタリングなどの手法により文を絞り込み，候補として残った文のそれぞれと入力文との 2 文間の照合を繰り返す．例えば文献 [13] の方法では，与えられた文との類似度の大きな重心を持つクラスタを選ぶことにより検索対象を絞り込む．このシステムでは検索もれが生じうる．つまり選択されなかったクラスタ中に入力文に最も類似した文が存在する可能性がある．また翻訳メモリに関する他の研究 [34] では品詞レベルでの文の完全一致に基づいた検索を提案している．この方法では単語の挿入，削除や意味的類似度を考慮した柔軟な照合に対応できない．本論文では，用例の集合の中から編集距離最小の用例を探索する問題の解法として，効率的で検索もれのない手法を提案する．この手法では 2 文間の逐次的な照合は行なわず，グラフ化された複数の候補文と入力文との照合を同時並行的に進める．

2.2 編集距離を使った用例翻訳

本論文では対象とする文単位の用例翻訳システムとして，編集距離に基づく類似度を使ったシステム（DP-match Driven transDucer，以下，D^3 と記す）[40] を想定する．以下 D^3 の概要について説明する．

2.2.1 構成

図 2.1 に示すように，D^3 は対訳コーパス，対訳辞書，シソーラスの 3 種類の言語資源を使用する．対訳コーパスは，翻訳対象の原言語と目的言語の文の対の集合であ

る．両言語の文は互いに対訳関係にある．対訳コーパス中の文は単語に分割され品詞情報が付与されている．D^3 では，対訳コーパス中の対訳関係にある文のペアを利用して翻訳を実行する．以下，この文のペアを用例と呼ぶ．対訳辞書は翻訳パターン抽出と訳語置換処理で使われる．シソーラスは，原言語と目的言語それぞれに用意され，用例検索および翻訳パターン抽出処理で使われる．

2.2.2　翻訳処理

同じく図 2.1 に示すように，翻訳処理は (1) 用例検索，(2) 翻訳パターン抽出，(3) 翻訳パターン選択，(4) 訳語置換，の 4 フェーズからなる．以下，各フェーズについて説明する．

用例検索

用例検索処理は全用例の原言語文を走査する．入力文と用例原言語文の単語列間の距離を測り，最小距離の用例を選び出す．この最小距離が大きければ検索された用例は翻訳処理に有用ではない．そのため距離に閾値を設ける．閾値以内の距離の用例が存在しなければ用例検索および翻訳処理は失敗に終わる．単語列間の距離には意味距離の加味された編集距離を使う．この編集距離 $dist$ は式 (2.1) で表される．この式で，$|s_i|$ は入力文の単語数，$|s_e|$ は用例原言語文の単語数，I は挿入単語数，D は削除単語数，Sem は置換語の意味距離を示す．この式に従って，挿入語と削除語の数，置換語の意味距離が足し合わされ，入力文と用例原言語文の長さの和でもって正規化される．置換については，2 単語が同品詞の内容語である場合のみ対象となり，2 単語間の意味距離が編集距離の計算に使われる．置換のコストには Sem で示される 2 単語間のシソーラスに基づく意味距離が使われる．Sem は，式 (2.2) で示すように，K(シソーラスにおいて 2 単語の範疇が異なる概念階層数) を N (シソーラスの概念階層数) で割った値である [42]．Sem は 0 以上 1 以下の値をとる．

$$dist = \frac{I + D + 2\sum Sem}{|s_i| + |s_e|} \tag{2.1}$$

$$Sem = \frac{K}{N} \tag{2.2}$$

次に日英翻訳での例を示す．(1-j) は入力文，(2-j) は距離の最も小さい用例の原言

語文とする．網かけ部分が両文の差分となる．

(1-j) 色/が/気/に/入り/ません
(2-j) デザイン/が/気/に/入り/ません

ここで「色」と「デザイン」がシソーラス上で完全に異なった語とすると単語間の意味距離は 1 となり，2 文間の編集距離は，(0+0+2*1)/(6+6)=0.167 となる．この編集距離は DP マッチと呼ばれる動的プログラミングの手法により計算可能である [12].

翻訳パターン抽出

検索された用例の原言語文中の入力文と異なる箇所を変数とし，用例目的言語文中の対応する箇所に同じ変数を当てはめた翻訳パターンを生成する．両言語の文の間で対応をとる際は変数となる単語のみ対象とし，全ての単語の対応をとる必要はない．つまり変数部分以外の箇所は全体として対応していると仮定する．このため用例のほとんどの部分は変更されず，訳文の組み立て時に発生する誤りや不自然さの回避が期待される．この原言語と目的言語の単語間の対応をとるには様々な単語アラインメントの手法 [27] の適用が考えられる．今回の実装では対訳辞書，両言語のシソーラスに基づいて単語間の対応関係を判断している．

先の例の (2-j) に対応する目的言語文を (2-e) とする．このフェーズでは (2-e) 中で「デザイン」に対応する箇所が探しだされ，文中の網かけ部分で示された対応がとられる．

(2-j) デザイン/が/気/に/入り/ません
(2-e) I do not like the design.

この結果，原言語パターン (2-j-p) と目的言語パターン (2-e-p) からなる翻訳パターンが作られ，入力文による変数束縛が (1-j-b) となる．

(2-j-p) X/が/気/に/入り/ません
(2-e-p) I do not like the X.
(1-j-b) X=「色」

翻訳パターン選択

最小距離の用例が複数あり複数種類の翻訳パターンが作られる場合がある．複数の翻訳パターンから一つを選択するためには，1) 入力文とパターンの不整合の小さい方を選ぶ，2) より多くの用例検索結果から同じ翻訳パターンが抽出された方を選ぶ，3) 翻訳パターン中に現れる単語のコーパスでの出現頻度の合計が大きい方を選ぶ，というヒューリスティクスを使用する．1) の不整合は，目的言語文と対応のとれない変数の個数，挿入語数，削除語数が大きいほど大きくする．これらで決定できない場合は任意の一つの翻訳パターンを選ぶ．

訳語置換

翻訳パターンの変数に束縛された単語の訳語を対訳辞書[*1]から引き，その訳語でもって目的言語パターンの変数を具体化する．

先の例を使うと目的言語側の変数束縛は (1-e-b) となり，訳文 (1-e) が得られる．

(1-e-b)　X=”color”
(1-e)　I do not like the color.

挿入と削除

ここまでの説明に用いた例では，用例の原言語文と入力文とで異なる部分同士の対応をとっているが，本方式では対応をとることのできないパターンも許している．用例側に対応する語を持たない入力文側の語，すなわち挿入語は，翻訳パターンに組み込まれず訳語置換フェーズでは無視される．入力文側に対応する語を持たない用例側の語，すなわち削除語は，束縛されない変数となる．目的言語パターン中に同じ変数があれば，その変数欄には訳語置換処理によって空文字列が入れられ，結果として用例目的言語文中にあった語が翻訳文から削除されることになる．目的言語パターン中に同じ変数がなければ，削除語は生成される翻訳文に影響を与えない．

挿入や削除により入力文や用例からの情報の欠落が起こりうるが，単語列間の編集距離の定義により，意味的に類似した語の置換が挿入や削除よりも優先される．その上で挿入や削除を許すことにより，特に話し言葉における助詞の脱落などの現象に対

[*1] 今回の実装では訳語選択はしていない．一つのエントリに対する訳はただ一つである．

する頑健性を与える．

例えば用例原文「コーヒー/か/紅茶/は/いかが/です/か」に対し，入力文「ビール/か/ワイン/は/いかが/です/か」では 2 箇所の置換があるが，両置換において対応する語は飲み物としてシソーラス上一致し，意味距離さらには編集距離は 0 に近い値となる．また同じ用例原文に対して，「コーヒー/か/紅茶/いかが/です/か」という入力文が与えられると，係助詞「は」一語の削除が発生し編集距離は $1/(6+7)=0.077$ となる．より近い用例が見つからなければ，この用例が訳文生成に利用される．逆に用例側の助詞が省略されている場合は助詞の挿入が発生することになる．

第 3 章

グラフベースの検索手法

3.1 用例検索

前章で説明した D^3 の各処理の中で翻訳実行時間の大きな割合を占めるのは用例検索である．用例の選択基準には 2.2.2 節で定義された単語列間の編集距離が使われる．用例検索処理は，用例の原言語文を候補文とし，入力文との距離が閾値以内で最小の候補文を全て求めることである．この編集距離は 2 文間の関係で定義されていて，二つの単語列の DP マッチにより計算可能である．従って各候補文と入力文間の DP マッチを逐次的に繰り返すことで最小距離の候補文を求めることができる．しかし単純にこの方法を使うとすれば用例数に比例した処理時間がかかってしまい，大規模用例を利用したリアルタイムの翻訳処理を実現することは通常の計算機では難しい．そこで我々は，候補文集合の分割，単語グラフ，A*アルゴリズムを利用した効率的な実装を提案する．この実装方法では編集距離の定義と与えられた閾値に関して検索もれはない．つまり DP マッチを全候補文に対して逐次的に繰り返した場合と同じ距離最小の候補文の集合を検索結果として返す．

3.1.1 候補文集合の分割

内容語数と機能語数を基に候補文をグループ分けする．これにより入力文の内容語数と機能語数および距離閾値により検索対象の候補文数を絞ることができる．つまり，機能語同士，内容語同士はすべて一致すると仮定した場合のグループ毎に可能な最小距離を求める．最小距離が距離閾値の範囲内で小さいグループから順に検索す

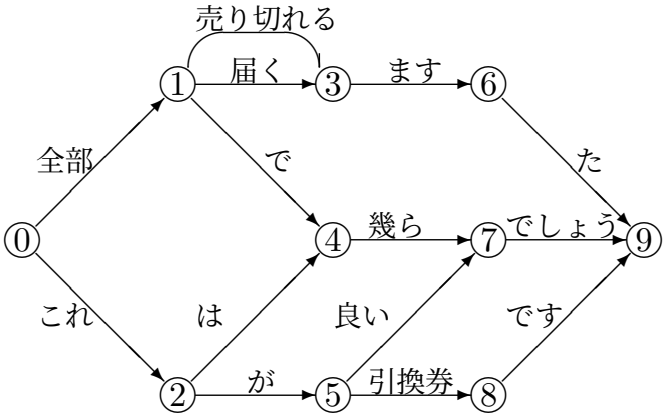

図 3.1. 単語グラフの例

る．あるグループ中に解が見つかれば，その解の距離を新たな閾値として検索対象のグループはさらに絞られる．また，グループ内での処理については 3.2 節で説明するが，グループ内では全ての候補文の内容語数と機能語数が等しい，つまりは単語数も等しいということを前提条件として利用する．

3.1.2 単語グラフ

内容語数と機能語数を基準に分けられたグループ毎に複数の候補文が一つの単語グラフにまとめられる．図 3.1 に単語グラフの例を示す．単語グラフは有向グラフであり，先頭ノードから最終ノードに至る可能な道筋と候補文が互いに対応する．複数の文に共通な単語列がグラフ中で一つにまとめられ，ノード数が最小となるように圧縮されている．この圧縮処理には有限状態オートマトンを変換する手法 [9] が使われている．単語グラフを利用することにより，グループ内の全候補文を同時並行的に調べながら，入力文との距離が最小の候補文を検索する．

3.1.3 A*アルゴリズム

二つの単語列を照合した結果を示す単語の一致，置換，挿入，削除の列を照合列と呼ぶこととする．グループ内の検索は，単語グラフの先頭ノードから最終ノードまでの可能な全経路について，各経路に現れる単語列と入力単語列との照合列の中から単語列編集距離を最小にするものを探索することである．この探索問題の解法に A*アルゴリズムを用いる．一般に A*アルゴリズムでは，問題状態集合の中から最終コス

トの下限の推定値が最小のものが選ばれ継続状態に展開される．ここで対象とする問題では，状態は，単語グラフの経路と入力文との照合の途中経過を意味する．

3.2　探索

本節では一つの単語グラフを使った探索処理に絞って説明する．単語グラフはノードとリンクからなる．リンクは単語をラベルとして持ち一つの始点ノードと一つの終点ノードを結ぶ．単語グラフ全体で一つの先頭ノードと一つの最終ノードを持つ．

3.2.1　状態空間表現

対象となる問題の状態空間は以下で説明する状態，作用素，初期状態と目標状態により構成される．

状態

状態は paths，node，input，trans の属性を持つ．各属性の内容は以下の通りである．

- paths: その時点までの照合列のリスト．
- node: 単語グラフのノード．このノードまで照合が進んだことを示す．
- input: 入力単語列の未だ照合に使われていない部分．
- trans: 適用可能な作用素を示す．

paths 内の各照合列の一致，置換，挿入，削除をそれぞれ，(E, 単語)，(S, グラフ側単語, 入力側単語)，(I, 入力側単語)，(D, グラフ側単語) の形式で表し，それぞれ E レコード，S レコード，I レコード，D レコードと呼ぶ．状態のコストは paths 内の任意の一つの照合列のコストである．paths 中のどの照合列も等しいコストを持つ．照合列のコストは，それに含まれるレコードのコストの和であり，各レコードのコストは E レコードは 0，I レコードは 1，D レコードは 1 と定義される．S レコードのコストは 2 単語間の意味距離を 2 倍した値であるが，意味距離が 0 の場合には小さな正の値を与える[*1]．この値が S レコードの最小コストとなる．

[*1] これは類似語関係と同一語関係を区別するためである．

作用素

状態に作用素を適用することにより継続状態が生成される．一般に一つの状態に複数の作用素が適用可能であり，一つの状態からいくつかの継続状態が生成される．下で 5 種類の作用素を定義する．T 作用素と I 作用素は状態に適用されるが，E，S，D の各作用素は状態，および，その状態の node を始点とするリンクの組に適用される．以下の説明では，作用素の適用される状態を s，リンクを l，生成される継続状態を s' と表し，各作用素について適用条件とどのような継続状態が生成されるかを記述している．

- T 作用素:
  - 条件: s.trans が E 作用素または S 作用素である．
  - 生成: s'.trans = s.trans が E 作用素ならば S 作用素と NIL から選択（説明は後述），s.trans が S 作用素ならば NIL
    s' の他の属性値は s と同じ
- E 作用素:
  - 条件: s.trans が E 作用素である．
    かつ，s.input が空リストでない．
    かつ，l のラベルと s.input の先頭が同一語である．
  - 生成: s'.paths = s.paths の各要素に E レコードを追加した値
    s'.node = l の終点
    s'.input = s.input から先頭を消去した値
    s'.trans = E 作用素と S 作用素と NIL から選択（説明は後述）
- S 作用素:
  - 条件: s.trans が S 作用素である．
    かつ，s.input が空リストでない．
    かつ，s.input の先頭と l のラベルが同品詞の内容語であり，かつ，同一語ではない．
    かつ，これら 2 単語の意味距離は 1 未満である．
  - 生成: s'.paths = s.paths の各要素に S レコードを追加した値
    s'.node = l の終点
    s'.input = s.input から先頭を消去した値

s'.trans = E 作用素と S 作用素と NIL から選択

- I 作用素:
  - 条件: s.trans が NIL である.
    かつ，s.input が空リストでない.
  - 生成: s'.paths = s.paths の各要素に I レコードを追加した値
    s'.node = s.node
    s'.input = s.input から先頭を消去した値
    s'.trans = E 作用素と S 作用素と NIL から選択
- D 作用素:
  - 条件: s.trans が NIL である.
    かつ，s.paths に最新レコードが I レコードでない要素がある.
  - 生成: s'.paths = s.paths から最新レコードが I レコードである要素を除き，残った要素に D レコードを追加した値
    s'.node = l の終点
    s'.input = s.input
    s'.trans = E 作用素と S 作用素と NIL から選択

上記で「S 作用素と NIL から選択」とは，s' に S 作用素を適用できる可能性があれば s'.trans の値を S 作用素とし，可能性が無ければ NIL とすることである．ここでは，s'.input の先頭が内容語であり，その語との同一語を除く同品詞語をラベルとし s'.node を始点とするリンクが存在する場合に S 作用素を適用できる可能性があると判断する．また「E 作用素と S 作用素と NIL から選択」とは，s'.input の先頭語をラベルとし s'.node を始点とするリンクが存在すれば s'.trans の値を E 作用素とし，そうでなければ S 作用素と NIL から選択する．T 作用素は実際に照合を進める作用素ではなく，trans 属性とともに E，S，I，D の各作用素の適用順序を制御する役目を持つ．D 作用素の 2 番目の条件は I レコードの後に D レコードが来るのを防いでいる．つまり，I レコードと D レコードが連続する場合 D が先に来るようにし，実質的に同じ照合列が複数現れる冗長性を排除する.

初期状態と目標状態

初期状態では paths は空リストを要素とするリスト，node は先頭ノード，input は入力単語列全体，trans は E 作用素となる．目標状態は，node が最終ノード，か

つ，input が空リストであるような状態である．

3.2.2 探索アルゴリズム

上記の初期状態，作用素，目標状態で表現される状態空間からコスト最小の目標状態を探索する．初期条件としてコスト上限値が与えられる．コスト上限値は入力文長と候補文長の和を距離閾値に乗じた値である．

評価関数

状態空間探索時に使用する評価関数 f^* を次のように定義する．

$$f^*(s) = g(s) + h^*(s) \tag{3.1}$$

$g(s)$ は初期状態から状態 s に達するまでにかかったコスト，つまり先に定義した状態のコストであり s.paths から計算できる．目標状態では $f^*(s) = g(s)$ となる．$h^*(s)$ は状態 s から目標状態までにかかるコストの下限である．一つの単語グラフを構成する全候補文の内容語数，機能語数はそれぞれ同一であるため，状態 s において入力文側とグラフ側の未処理の内容語数，機能語数は一意に決まる．それぞれの個数を C_{input}，C_{graph}，F_{input}，F_{graph}，として，残り語数に基づく最小コスト $h'(s)$ を次のように計算する．

$$h'(s) = |C_{input} - C_{graph}| + |F_{input} - F_{graph}| \tag{3.2}$$

さらに，T 作用素の適用が先行する場合を含めて，状態 s に最初に適用可能な E, S，I，D の各作用素について，それが適用されたと仮定したときの目標状態までにかかるコストの下限を次の値とする．

- E 作用素:
  $h'(s)$
- S 作用素:
  $h'(s)$ に S レコードの最小コストを加えた値.
- I 作用素:
  s.input の先頭が内容語の場合は, $|(C_{input} - 1) - C_{graph}|+|F_{input} - F_{graph}|+1$，機能語の場合は，$|C_{input} - C_{graph}| + |(F_{input} - 1) - F_{graph}| + 1$.

- D 作用素:
  s.node を始点とするリンクのラベルが内容語のみであれば，$H_c + 1$,
  s.node を始点とするリンクのラベルが機能語のみであれば，$H_f + 1$,
  いずれでもなければ，$H_c + 1$ と $H_f + 1$ のうち小さい方の値．ここで，
  H_c=$|C_{input} - (C_{graph} - 1)| + |F_{input} - F_{graph}|$,
  H_f=$|C_{input} - C_{graph}| + |F_{input} - (F_{graph} - 1)|$.

これらを使って $h^*(s)$ を，1) s.trans が E 作用素のときは，E 作用素が適用されたときのコストの下限，2) s.trans が S 作用素のときは，S 作用素，I 作用素または D 作用素が適用されたときのコストの下限の最小値，3) s.trans が NIL のときは，I 作用素または D 作用素が適用されたときのコストの下限の最小値，とする．

アルゴリズム

探索アルゴリズムは下のように表される．この記述中，OPEN は未展開状態を，CLOSED は展開済み状態を保持するためのリストを示す．また 5. における「同じ状態」とは，paths を除く属性値の等しい状態を意味する．

1. コスト上限を与えられた値にする．OPEN に初期状態のみを入れる．
2. OPEN にコスト上限以下の状態がなければ終了．
3. OPEN から評価関数 f^* を最小にする状態 s を取り除き，CLOSED に入れる．
4. s が目標状態なら，それを解の一つとし，コスト上限を s のコストに変え，2. に戻る．
5. s の全ての継続状態を生成する．各継続状態 s' について，$f^*(s')$ がコスト上限以下であれば，OPEN および CLOSED 中の同じ状態と比較し，
   （a）同じ状態がなければ，s' を OPEN に追加．
   （b）s' よりコストの大きい同じ状態が OPEN または CLOSED に既存であれば，この既存状態を消去し，s' を OPEN に追加．
   （c）s' とコストの等しい同じ状態が CLOSED に既存であれば，この既存状態を消去し，s' を OPEN に追加．
   （d）s' とコストの等しい同じ状態が OPEN に既存であれば，この既存状態の paths に s'.paths をマージする．
6. 2. に戻る．

単語グラフの特徴の利用

単語グラフの形状の特徴として，先頭ノードを始点とするリンク数は他のノードを始点とするリンク数よりも圧倒的に大きくなる傾向がある．そのため node 属性に先頭ノードを持つ状態に D 作用素が適用されると多くの継続状態が作られることとなり計算時間が大きくなる．これは照合列の先頭要素が D レコードとなる場合である．この展開数の増大を避けるため，単語グラフ中，先頭ノードから数段階の仮のリンクとノードを加える．先頭ノードを持つ状態から D 作用素によって第 1 の仮のノードを持つ状態へ遷移する．第 1 の仮のノードは，全候補文について 2 番目の語をラベルとするリンクの始点となり，通常の単語グラフのノードに合流する．第 1 の仮のノードにある状態は E 作用素または S 作用素の適用により通常のノードの状態に，D 作用素によって第 2 の仮のノードを持つ状態に遷移する．

何段階まで仮のノードを用意するかは，用例検索時に使われる可能性のある距離閾値の最大値から計算できる．候補文の長さを L とすると，照合列の先頭に D レコードが d 個並ぶという条件で，候補文との距離を最小にする入力文は，候補文から先頭 d 語を除いた文である．そのときの距離は $d/((L-d)+L)$ である．この値が距離閾値の最大値を越える場合は探索する必要がない．距離閾値の最大値を Θ とすると $d/((L-d)+L) \leq \Theta$ から $d \leq 2\Theta L/(1+\Theta)$ が導かれる．この式を満たす d の最大の整数値が用意すべき仮のノードの段数である．

3.2.3 実行例

探索の実行例を示す．図 3.1 の単語グラフから入力文「全部揃いました」の類似文を検索することにする．ここでは状態を [paths, node, input, trans, f^* 関数値] の形式で記述する．node 値には 図 3.1 中でノードに付けた番号を用いる．δ を S レコードの最小コストとする．また「揃う」と「売り切れる」の意味距離を 1.0,「揃う」と「届く」の意味距離を 0.7 であると仮定する．

初期状態 $s0$ は次のようになる．

$s0 =$ [((()), ノード 0, (全部, 揃う, ます, た), E 作用素, 0]

$s0$ に適用可能な作用素は E 作用素と T 作用素である．これらの作用素を適用して継続状態 $s1$ と $s2$ が得られ，OPEN は $\{s1, s2\}$ となる．ノード 0 を始点とするリンク

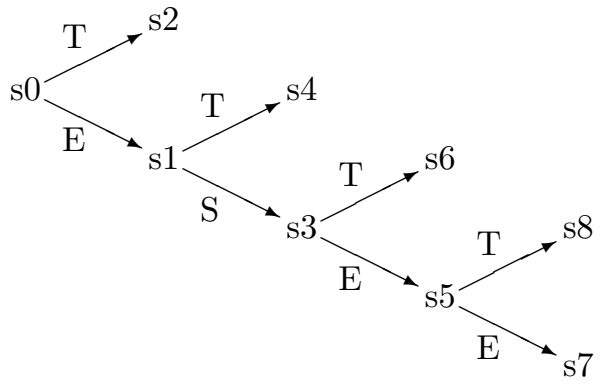

図 3.2. 状態遷移図

のラベルには同一語を除いて「全部」の同品詞語がないため $s2$.trans は NIL になる．

$s1$ = [(((E, 全部))), ノード 1, (揃う, ます, た), S 作用素, δ]
$s2$ = [(()), ノード 0, (全部, 揃う, ます, た), NIL, 2]

ここで OPEN から f^* 関数値の小さな $s1$ が選ばれ展開される．$s1$ に適用可能な作用素は S 作用素と T 作用素である．「売り切れる」と「届く」をラベルとする二つのリンクに関して S 作用素の適用条件がテストされるが，1 未満の意味距離の得られる「届く」のリンクについてのみ成功する．S 作用素の適用により継続状態 $s3$ が得られる．また T 作用素の適用により継続状態 $s4$ が生成される．OPEN は $\{s2,\ s3\ \ s4\}$ となる．

$s3$ = [(((E, 全部), (S, 届く, 揃う))), ノード 3, (ます, た), E 作用素, 1.4]
$s4$ = [(((E, 全部))), ノード 1, (揃う, ます, た), NIL, 2]

ここで OPEN から f^* 関数値が最小の $s3$ が選ばれ展開される．$s3$ に E 作用素を 2 回適用した $s7$ が解となる．

$s7$ = [(((E, 全部), (S, 届く, 揃う), (E, ます), (E, た))), ノード 9, (), NIL, 1.4]

この例の状態遷移の様子を 図 3.2 に示す．T 作用素の適用による遷移を除き解に向かって一直線に探索が進んでいる．

第 4 章

グラフベースの検索手法を使った翻訳システムの評価

前章で詳述したグラフベースの検索手法を用いて文単位の用例翻訳システム D^3 を実装し，大規模コーパスを使った日英翻訳実験によりその性能を評価した．

4.1 実験条件

本実験で用いた対訳コーパスは旅行会話基本表現集 (Basic Travel Expression Corpus，以下，BTEC と記す) [44] である．このコーパスは，海外旅行者向けのフレーズブックでよく見られるような表現を集めたものであり，日本語とその英訳が文単位で対応している．このコーパスのサイズを表 4.1 に示す．実験では日英 152,170 文ずつからなる学習セット[*1]と 510 文の日本語からなるテストセットを抜き出して使用した．

対訳辞書とシソーラスは，他の翻訳システム [43] 用に準備された旅行会話ドメインの既存のものを使用した．使用したシソーラスの階層構造は角川類語新辞典 [53] に準拠している．

翻訳結果の品質評価のためには，客観評価指標と主観評価指標を用いた．客観評価指標には多重参照単語誤り率 [45] （multi-reference Word Error Rate，以下，mWER と記す）と BLEU スコア [32] を使用した．mWER は翻訳結果が参照訳と

[*1] D^3 は学習を行わず与えられたコーパスをあるがままに利用するが，翻訳のための知識源を，他の用例翻訳と同様，学習セットという呼ぶことにする．

表 4.1. コーパスサイズ

	日本語	英語
文数	404,022	404,022
総語数	2,870,280	2,473,711
平均文長	7.10	6.12
語彙数	33,288	22,378

表 4.2. 日英翻訳性能

mWER	BLEU	主観評価指標 (%)			出力率	処理時間 (秒)	
		A	AB	ABC	(%)	平均	最大
0.3425	0.6399	71.2	80.4	83.3	91.8	0.218	3.310

異なる割合を編集距離に基づいて計算する．BLEU は翻訳結果と参照訳の N グラムの一致する割合を示す．翻訳結果の品質が高くなれば mWER は小さくなり，BLEU スコアは大きくなると考えられている．mWER, BLEU ともに各テスト文につき 16 文の正解翻訳例を参照訳とした．主観評価では，原言語を理解できる目的言語のネイティブにより 4 段階の評価を与えた [43]．評価ランクは，(A) 問題なし，(B) 主要な情報が容易に復元できる，(C) 主要な情報がかろうじて復元できる，(D) 主要な情報が復元できない，としている．主観評価指標としては，良い方のランクからの累計数の全テスト文数に対する割合を A，AB，ABC と表記して使う．ABCD は出力率として示す．

各実験での D^3 の距離閾値は特に触れない限り 1/3 とした．また実験には Pentium4/2GHz のコンピュータを使い，プログラムは Allegro Common Lisp 6.2 版上で実装，実行した．

4.2　基本的な翻訳性能

翻訳結果の評価を表 4.2 に示す．この表から A ランクの割合が大きく品質の高い翻訳文が得られていることが分かる．主観評価ランク別の翻訳結果の例を表 4.3 に挙

表 4.3. 主観評価ランク別の翻訳例

評価	入力文 → 翻訳結果
A	このあたりにデパートはありますか → Is there a department store near here?
B	頭痛がしますアスピリンはありますか → I have a headache. Can you recommend a aspirin?
C	三十ドル前後でおすすめのワインは何ですか → Do you know a good wine?
D	ただいま釣り銭を切らしております → He isn't available.

げておく[*2]．一方で出力率は 91.8% であり約 8% の文で翻訳結果が出力されていない．翻訳処理時間は，平均 0.2 秒，最大 3.3 秒であり，提案した用例検索手法により効率的な処理が実現されている．なお用例検索処理のために作られた単語グラフの個数は 335 であり，一つの単語グラフ内の平均ノード数は 728，平均リンク数は 996 であった．翻訳実行時にアクセスされる単語グラフ数は 1 入力文あたり平均 13.7 であった．

4.3 人間の英語能力との比較

この翻訳品質を人間と比較するとどのレベルに相当するか，どのレベルの英語能力を持った日本人が同等の翻訳文を作れるのかを文献 [54] の方法で求めた．この方法では色々なレベルの TOEIC スコア（Test Of English for International Communication）を持つ人間の作った翻訳文とシステムの翻訳出力を一対比較法で評価し，回帰分析により人間とシステムの能力の均衡する TOEIC スコアを計算する．

今回は TOEIC スコア 420 点から 965 点までの 5 人の日本人と比較した．表 4.4 に一対比較の結果を示す．ここから D[3] の能力として 870 点の TOEIC スコアが得られた．日本企業で海外と取引のある部門の社員の平均的なスコアが 750 点，860 点以

[*2] 本論文では翻訳入力に句読点を使わない．

表 4.4. D^3 と人間の一対比較

TOEIC スコア	D^3 の勝ち	人間の勝ち	引き分け
420	224	133	153
540	214	149	147
685	211	150	149
820	176	157	177
965	141	173	196

上で非英語ネイティヴとしては最高レベルと能力評価されていることから見ると，この実験では D^3 は高い品質を示したと言える．

4.4　他の文単位の用例翻訳との比較

D^3 は文単位の用例翻訳として，入力文に十分類似した用例さえあれば高い品質の翻訳を出すことが期待される．本節では，D^3 の編集距離に基づく検索指標が翻訳品質に与える効果を，他の文単位の用例翻訳の用いる指標と比較評価する．編集距離指標がより効果的であれば，この指標を用いた効率的な検索手法の意義が確認される．比較対象とする検索指標は，内容語の集合と文の法と時制を用いる．

比較対象手法

比較対象として文献 [38] で提案された文単位の用例翻訳 (Example-Based Rough Translation，以下，EBRT と記す) の用いる検索手法を用いる．EBRT は入力文に対して意味的に等価な文をコーパスから検索する．ここで意味的に等価な文とは，入力文と主要な意味を共有し，かつ入力文に含まれない情報を持たない文として定義される．現実的には次の条件を満たす文を意味的に等価な文と見なす．1) 入力文と同じ法と時制を持つ，2) その文に含まれる内容語の全てについて，同一語または類義語が入力文に含まれる，3) その文に含まれる内容語の少なくとも 1 つが入力文に含まれる．ここで，類義語はシソーラスに基づき判断される．法と時制は，主に助詞と助動詞に関する表層形に基づいたヒューリスティクスにより区別される．入力文に対して意味的に等価な文が複数ある場合，同一内容語の個数，類義内容語の個数，同一機能語の個数，異なる機能語の個数を考慮して 1 つの文を選択する．

選ばれた意味的に等価な文の訳文から，類義語に対応する部分をその訳語で置き換えることにより，入力文に対する訳文を生成する．この対応箇所の発見と訳語置換は 2.2.2 節で述べた D^3 での相当する処理と同じである．

実験では，4.1 節で述べた学習セット，対訳辞書，シソーラスを使って実現した EBRT によりテストセットの翻訳を行った．法と時制を区別するルールは，実験で使うコーパス中の文などを考慮しながら人手で作成した．

２つの手法の差異

EBRT では個々の内容語と抽象化された法と時制を考慮して類似文を判定する．一方 D^3 では個々の単語とその並ぶ順序を考慮して類似文を判定する．D^3 においては，法や時制に相当する情報の比較も単語列の照合という統一的な仕組みの中で扱われていると考えることができる．

表 4.5 の例は 2 つの手法の典型的な差異を示す．この表は入力文，および，2 つの手法で検索された対訳用例とそれらを使って得られた翻訳結果を示す．D^3 は「カメラ」を「ネックレス」で,「保証書」を「鑑定書」で置き換えた文を検索している．これらの置換関係は単語列の照合により発見されている．一方，与えられたシソーラスでは「カメラ」と「ネックレス」は互いに類義語とならないため，EBRT はこの文を検索することができない．代わりに EBRT は意味的に等価な文として，入力文の含む内容語の一部を含んだ文を検索している．D^3 の翻訳結果は意味的に完全であるのに対し，EBRT の訳は大筋妥当だが，入力文中の「カメラ」に相当する情報が抜け落ちてしまっている．

翻訳品質評価

表 4.6 に D^3 と EBRT の評価結果を示す．BLEU を除く全ての指標が，D^3 は EBRT より高い翻訳品質であることを示している．D^3 は，その ABC の割合がより高いことにより，より大きなカバレッジを持つことが分かる．また A の割合つまり品質の高い訳の割合も EBRT より 12% 大きい．2 つのシステムは用例検索後の処理が同じであるため，両者の訳の違いは検索された文の違いにある．それゆえ実験結果は，翻訳品質の面から，D^3 の用いる検索指標は EBRT の指標に比べより効果的であることを示している．

表 4.5. 翻訳例: D^3 と EBRT によって検索された用例と翻訳結果 (J は日本語を，E は英語を示す)

入力文		J: この/カメラ/には/保証書/が/付い/て/ます/か
検索された用例	D^3	J: この/ネックレス/には/鑑定書/が/付い/て/ます/か E: Does this necklace have a certificate?
	EBRT	J: 保証書/が/付い/て/ます/か E: Does it come with a guarantee?
翻訳結果	D^3	E: Does this camera have a guarantee?
	EBRT	E: Does it come with a guarantee?

表 4.6. D^3 と EBRT の翻訳品質

	mWER	BLEU	主観評価指標 (%)			出力率
			A	AB	ABC	(%)
D^3	0.3401	0.6554	71.2	80.4	83.3	91.8
EBRT	0.3984	0.6725	59.2	69.0	75.1	91.4

4.5　翻訳例と誤り分析

表 4.7 に翻訳の成功例，表 4.8 に典型的な失敗例をいくつか示す．ここでは失敗例に注目する．

(1-a)，(1-b)，(1-c) では入力文に含まれない情報が翻訳結果に現れる現象が起こっている．(1-a) の翻訳に使われた用例は（メキシコに行くつもりです → I'm going to visit Mexico after leaving the U.S.）であり，この対訳が限られた場面でしか使えない特殊な対訳であることが原因で翻訳が失敗している．(1-b) での用例は（便名と時間を教えて下さい → What is the flight number and departure time?）であり，原言語と目的言語間の単語の対応付けに失敗している．すなわち「時間」と”time”が対応すると判断され”departure”が残ってしまっている．(1-c) での用例は（無農薬のオレンジはありますか → Do you have pesticide free oranges?）である．これも単語対応付けの失敗である．実験で使用している対訳辞書では「無農薬」と”pesticide

表 4.7. 翻訳の具体例

入力文 → 翻訳結果 (距離)
これの色違いありませんか
→ Do you have this in different colors? (0.077)
直し代九十九ドルいただきます
→ There is a ninety nine dollar repair charge. (0.111)
今朝朝食以外に何かサービスをお受けになりましたか
→ Did you have any other services besides breakfast this morning? (0.200)
仕事それとも観光
→ Are you on business or pleasure? (0.250)

free” との対応がとれず，シソーラスを使って「無農薬」と”pesticide” の対応がとられ”free”が残ってしまっている．なお，この例では原言語側で「赤」と「オレンジ」は色の意味で類似語と判断され対応がとられている．

(2-a)，(2-b)，(2-c) では入力文の情報が翻訳結果から欠落している．これらは用例原文に対する挿入語の扱いの問題である．(2-a) での用例原文は「聞きもらしたのですが」であり入力文の「お名前を」が無視されている．(2-b) での用例原文は「この小包を日本に送りたいのですが」であり入力文の後半が無視されている．この例では，入力文を 2 つの文に分割可能で，一方のみを翻訳したと考えることもできる．(2-c) での用例原文は「私はジャズが好きです」であり入力文の「ハウスや」が無視されている．

(3) は訳語の誤りである．対訳辞書に従って「名」の訳語が”name”になっているが，この場合不適切である．これは訳語選択の仕組みの導入により改善される問題である．

なお主観評価が D ランクとなった 43 文において，(1-a)，(1-b)，(1-c) のタイプの誤りが 26 文，(2-a)，(2-b)，(2-c) が 15 文，(3) が 8 文で観察される．同一文で異なるタイプの複数の誤りが現れる場合もある．これら三つのタイプの誤りのいずれも現れないものは 4 文ある．

個々の問題にはそれぞれの対策が考えられるが，いずれの問題も適切な用例の追加による改善の可能性がある．4.7 節の実験で学習セットを拡張したとき (2-b) の翻訳結果は”What is the quickest way to send this package to Japan?” に改善されて

表 4.8. 翻訳誤りの典型例

(1-a) ボストン美術館に行くつもりです
→ I'm going to visit Boston Museum after leaving the U.S.
(1-b) お名前とお部屋番号を教えて下さい
→ What is the name and departure room number?
(1-c) ワインの赤はありますか
→ Do you have wine free red?
(2-a) お名前を 聞きもらしたのですが
→ I missed that?
(2-b) この小包を日本に送りたいのですが 一番はやい方法は何ですか
→ I'd like to send this parcel to Japan.
(2-c) 私は ハウスや ソウルミュージックが好きです
→ I like soul music.
(3) 二 名 です禁煙席をお願いします
→ Just two name, nonsmoking section please.

いる．

4.6　翻訳カバレッジ

ここまで示したように，個々の翻訳結果には誤りもあるが，全体としては人間や他システムと比較して D^3 の翻訳品質は高い．これは表 4.2 の示す通り，A ランクつまり高品質の訳の割合が大きいことによる．一方，同じ表からは，ABC ランクつまり何らかの意味のある翻訳の割合が 83.3% に過ぎないことも分かる．つまり 16.7% の入力文に対して翻訳が失敗している．図 4.1 は編集距離と主観評価ランクとの関係を示す．この図から距離が大きくなると明らかに翻訳品質が悪くなることが分かる．つまり大きな距離の用例では満足な翻訳が不可能であり，このことが ABC ランクで示される翻訳カバレッジの低さにつながっている．

また表 4.9 に入力文の長さと翻訳品質の関係を示す．入力文が長くなると翻訳品質が悪くなることが分かる．一般に長い文ほど機械翻訳は難しくなる．特に D^3 では，複数の用例を組み合わせる仕組みを持たないので，長い文に弱く翻訳カバレッジの低

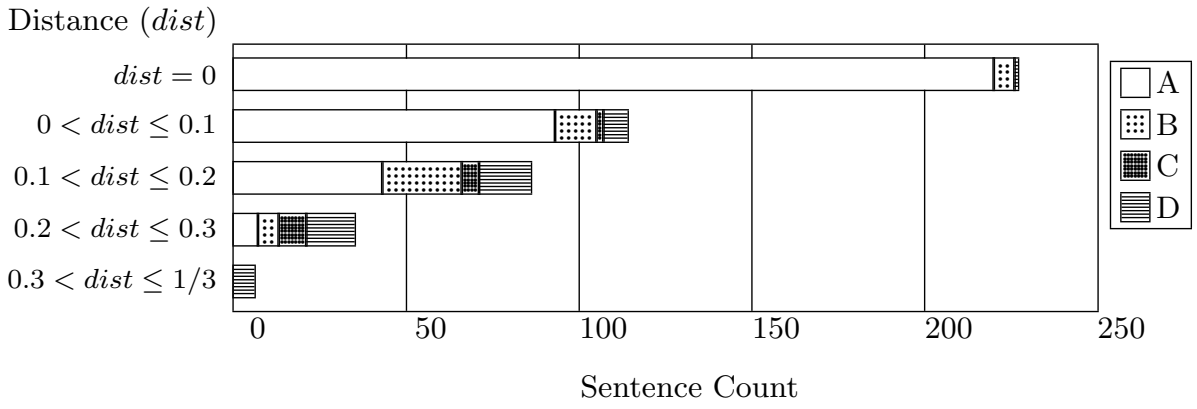

図 4.1. 編集距離と翻訳品質

表 4.9. 文長と翻訳品質

文長 (L)	文数	mWER	BLEU	主観評価指標 (%)			出力率
				A	AB	ABC	(%)
$L \leq 5$	186	0.2455	0.7029	86.6	89.8	90.9	98.4
$6 \leq L \leq 10$	274	0.3568	0.6351	67.5	81.0	83.6	92.3
$11 \leq L$	50	0.6001	0.4109	34.0	42.0	54.0	64.0

さの要因となる．ただし今回の実験に使ったテストセットでは長い文が少なかったため，影響は比較的小さく済んでいる．

カバレッジの低さは文単位の用例しか使わないシステムの本質的な欠点であり，このシステムは高品質であるが低カバレッジの翻訳方式と見ることができる．この問題に対するコーパスサイズの与える効果を次節で検証する．

4.7 大規模コーパスを使うことによる効果

翻訳カバレッジを広げ十分な翻訳品質を達成するために大規模コーパスを使うことの効果を検証した．コーパスサイズの影響を調べるため，4.1 節に示した学習セットのサイズを 2 倍，1/2，1/4，1/8 としたときの翻訳結果の品質および翻訳処理速度を評価する．図 4.10 に学習セットのサイズを日本語文ののべ数と異なり数で示す．テストセット 510 文の中で学習セット中の文と一致する文の数も示している．300K と表記した条件，つまり基の学習セットを 2 倍にした条件では，BTEC のうち基の学

表 4.10. 学習コーパスサイズ

コーパスサイズ		19K	38K	75K	150K	300K
のべ文数		19,022	38,043	76,085	152,170	304,340
異なり文数		15,923	29,785	54,657	97,116	199,664
一致テスト文 (のべ)	文数	104	142	181	227	240
	率 (%)	20.4	27.8	35.5	44.5	47.1

表 4.11. コーパスサイズ別日英翻訳性能（距離閾値=1/3）

サイズ	mWER	BLEU	主観評価指標 (%)			出力率	処理時間 (秒)	
			A	AB	ABC	(%)	平均	最大
19K	0.4617	0.5153	53.7	65.9	71.0	82.5	0.062	0.550
38K	0.4136	0.5824	58.2	70.6	75.1	86.3	0.085	0.930
75K	0.3858	0.6157	64.7	74.1	79.0	88.8	0.121	1.690
150K	0.3425	0.6399	71.2	80.4	83.3	91.8	0.218	3.310
300K	0.3213	0.6331	71.2	81.8	84.5	93.3	0.320	6.650

表 4.12. コーパスサイズ別日英翻訳性能（距離閾値=1/4）

サイズ	mWER	BLEU	主観評価指標 (%)			出力率	処理時間 (秒)	
			A	AB	ABC	(%)	平均	最大
19K	0.5080	0.4567	51.0	61.6	64.1	72.2	0.031	0.390
38K	0.4610	0.5268	55.1	65.9	68.8	77.6	0.041	0.680
75K	0.4174	0.6072	63.1	71.8	75.1	82.2	0.053	0.530
150K	0.3626	0.6608	70.2	78.4	80.2	86.5	0.096	1.600
300K	0.3364	0.6496	70.4	80.2	82.2	89.4	0.133	2.040

習セットやテストセットに使っていない部分の対訳を学習セットに追加した．基の学習セットより小さな学習セットを使う条件では，基の学習セットから無作為に対訳を取り除いた．

実験結果を表 4.11 に示す．カバレッジを示す ABC ランクの割合および他の指標から，コーパスサイズを大きくするにつれて翻訳品質が向上していることが分かる．

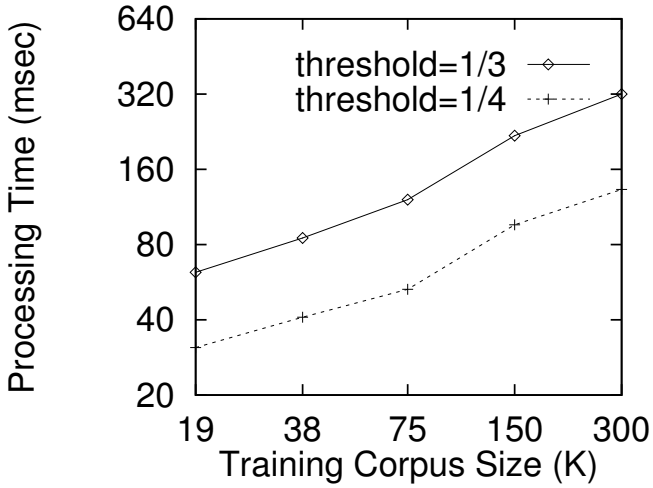

図 4.2. コーパスサイズと翻訳処理時間

主観指標と客観指標は大筋で一致した傾向を示す．

表 4.12 は距離閾値が 1/4 の場合の結果を示す．表 4.11 と比較すると，当然ながら翻訳出力率は下がるが，処理時間の短縮は明瞭である．閾値を小さくして用例を増やすと，処理時間の増大を抑え，かつ翻訳出力率を落とさないことが可能である．このとき翻訳結果の得られた入力文の用例に対する距離は同じか小さくなり訳質向上につながる．つまりコーパスサイズと距離閾値の組合わせによって訳質，出力率，処理速度が調整できる．例えば，表 4.12 のコーパスサイズ 300K の結果は，表 4.11 でのコーパスサイズ 75K の結果に比べ，10% ほどの処理時間の増加があるが，翻訳品質の高さとカバレッジの広さで上回っている．

図 4.2 はコーパスサイズと平均処理時間の関係を示す．縦軸，横軸は対数スケールとなっている．コーパスサイズを大きくするに従って処理時間も大きくなっている．しかし増加の程度は線形比例ではなく，処理時間はコーパスサイズのほぼ 1/2 乗のオーダーに抑えられている．

4.8 DP マッチとの比較

DP マッチを逐次的に繰り返す方法と提案検索手法の処理速度を比較した．比較対象として DP マッチを使った 3 手法を用意した．それぞれ Simple-DP，Class-DP，Pruning-DP と呼ぶこととする．Simple-DP では，まず入力文と同じ文をハッシュテーブルを使って探し，同じ文が無い場合に限り，全候補文について入力文との DP

表 4.13. DP マッチを使った手法との平均処理時間の比較 (処理時間をミリ秒単位で示す)

コーパスサイズ	19K	38K	75K	150K	300K
Simple-DP	2,752	4,815	8,101	12,731	26,189
Class-DP	1,286	2,233	3,813	6,045	10,925
Pruning-DP	539	880	1,449	2,310	3,961
提案手法	62	85	121	218	320

マッチを繰り返すことにより距離最小の文を見つける．ここで，DP マッチ処理の実行中は正規化しない距離，つまり式 (2.1) の分子をコストとして計算する．正規化された距離は DP マッチ後に求める．Class-DP は，Simple-DP を基にして，3.1.1 節で示した候補文集合の分割処理を加えた手法である．Pruning-DP は Class-DP にさらに改良を加えたものである．Pruning-DP では各候補文と入力文との DP マッチにおいて，その 2 文間の距離が，既発見の最小距離または距離閾値を超えると判断された時点で DP マッチ処理を終了させる．この判断は，計算済みコストおよび 2 文の未計算の単語数の差に基づく．これら 3 手法と提案手法は，与えられた入力文に対する検索結果として同じ候補文の集合を返す．実験では，距離閾値を 1/3 とし，各検索手法を使ったシステムによるテストセットの翻訳を行い処理時間を比較した．

表 4.13 は各手法を使ったときの平均処理時間を示す．Pruning-DP は実際 Simple-DP や Class-DP を上回る性能を示しているが，その Pruning-DP を大きく上回る性能を提案手法は示している．提案手法は Pruning-DP よりも，コーパスサイズが 19K のとき 8.7 倍，300K のとき 12.4 倍効率的である．

4.9　まとめ

文単位の用例翻訳は，入力文に十分類似した用例が存在すれば自然で高品質な訳文を生成可能であるが，本質的に低カバレッジである．この問題を緩和するためには大規模コーパスが必要であり，必然的に効率的な検索手法が必要となる．本章では提案検索手法を使って実装した用例翻訳システムを評価した．実験では旅行会話ドメインの数十万規模の文を持つ対訳コーパスを利用し，日英翻訳での翻訳品質と処理時間について検証した．結果として当該用例翻訳システムは，大規模コーパスを使うことに

より高品質な翻訳性能を持ち，提案検索手法を使うことにより効率的な処理が可能となっていることを確認した.

第 5 章

前処理による翻訳単位の適正化

機械翻訳システムの性能向上の目的のためには，翻訳エンジンの工夫だけでなく，前処理によって入力文を翻訳しやすい形に変換するアプローチもある．このアプローチの長所として，様々な翻訳エンジンに適用可能である点が上げられる．本章では，長い入力文に対する翻訳システムの精度を上げる課題に取り組む．文単位の用例翻訳システムの実験に関する 4.6 節の分析では，入力文が長くなると翻訳品質が悪くなっている．この問題は程度の差こそあれ機械翻訳システム一般にあてはまる．この問題に対処するため，前処理により入力文を翻訳しやすい単位に分割する手法を提案し，その効果を検証する．ここでは対象としてコーパスベース翻訳システムを使った対話文翻訳を想定している．

5.1 入力分割による翻訳精度向上の可能性

機械翻訳システムの性能を向上させるために，システムへの入力を分割し，分割した部分毎に翻訳するという方法は有望である．機械翻訳システムは翻訳誤りを犯すことがあるが特に入力が長くなると誤りが多くなる．入力が長くて翻訳誤りが起こる場合でも，分割した各部分に対しては翻訳が成功する可能性がある．通常機械翻訳システムへの入力単位は文であり，本論文でも入力文を翻訳するシステムを想定する．話し言葉翻訳における機械翻訳システムへの入力文は発話から切り出される．この場合入力文は複数の独立した文に分割可能な場合もある．特に対話においては，たとえ長い文であっても，複雑に入り組んだ文構造を持つことは少なく，互いに独立性の高い部分に分割可能な場合が多い．そのため，入力文の分割方法および分割された各部分

This is a medium size jacket / I think it's a good size for you / try it on please

図 5.1. 入力文の分割例 (/が分割点を示す)

の翻訳が適切であれば，各部分の翻訳結果を並べると入力全体に対する適切な翻訳となる可能性が高い．例えば，“This is a medium size jacket I think it's a good size for you try it on please” [*1]という入力は，“This is a medium size jacket”，“I think it's a good size for you”，“try it on please” の 3 つの部分に分割することができる．この場合，この 3 つの部分を翻訳しその結果を同じ順番に並べることにより，入力全体を翻訳することができる．本章でのアプローチは機械翻訳システムを使ったより良い翻訳が可能となるように入力文を分割することである．入力文の分割例を図 5.1 に示す．

文分割に関する従来の研究では，分割点周辺の N グラムなどの単語の連接の特徴に基づいた手法が多い [25, 5, 56, 52, 20]．これらの研究の中には文境界などを正解分割点にした場合の再現率や適合率で高い性能を上げているものもある．しかしながらこれらの分割点によって得られた分割部分に対して機械翻訳システムが良い翻訳結果を生成可能とは限らない．これらの従来研究では，考慮する素性や単語数などの詳細，利用する機械学習手法などに違いはあっても，分割点周辺のいくつかの単語を手がかりに分割点を決めることでは一致している．この考え方は自然であるが別の観点から改良する余地が残っている．

本論文では，単語連接に基づく分割を補うための別指標として類似度を導入する．提案分割手法では，N グラムに基づき分割点の候補を生成し，類似度に基づき分割点の最良の組合せを選択する．この選択は，コーパスベース翻訳システムは，学習コーパスに存在する文と類似した文は正しく翻訳することができるという仮定に基づいている．

以降の節では，提案分割手法の詳細について述べ，用例翻訳システムと統計翻訳システムを用いた実験結果を示し，類似度の導入による翻訳品質への効果を評価する．

[*1] 本論文では翻訳入力にピリオドやコンマなどの区切記号を使わない．

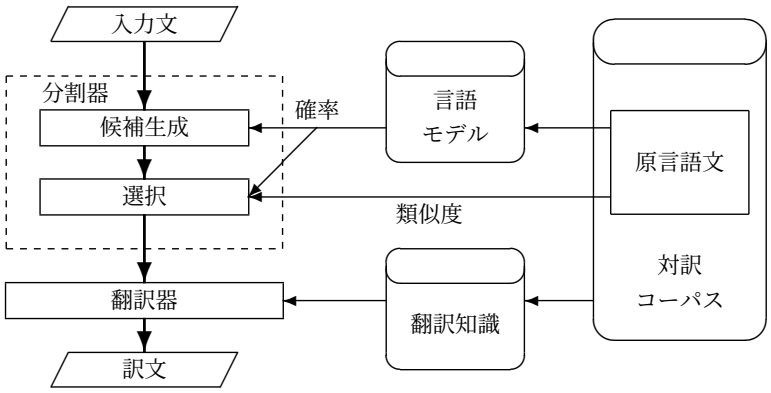

図 5.2. 文分割手法の構成

5.2　N グラム言語モデルと類似度を用いた分割手法

5.2.1　分割文表現

機械翻訳システムの入力文は単語列で表すものとする．文を分割して得られる任意の部分単語列もまた文とし，その部分単語列を基の文の部分文と呼ぶ．文を分割した結果として部分文の列が得られるが，その得られた列を分割文と呼ぶこととする．分割文中の部分文を連結すると分割前の基の文となる．このように分割文は文の列として表現される．分割文は要素として 1 つまたは複数の文を含む．入力文自体は 1 つの要素からなる分割文と見ることができる．5.1 節で使った例では，入力文自体を分割文として表現すると [“This is a medium size jacket I think it’s a good size for you try it on please”] となり，3 分割した結果は [“This is a medium size jacket”, “I think it’s a good size for you”, “try it on please”] となる．

5.2.2　構成

提案分割手法は，N グラム言語モデルを使っていくつかの分割文を候補として生成し，N グラム言語モデルと類似度を使って最良の候補を選択する．提案分割手法の構成を図 5.2 に示す．機械翻訳システムの翻訳知識は対訳コーパスから得られる．対訳コーパスでは原言語部分と目的言語部分が文単位で対応している．分割手法は同じコーパスの原言語部分を利用して言語モデルの学習や類似度計算を行う．

5.2.3 N グラム言語モデルに基づく指標

与えられた単語列が文である確率を求めるためのモデルとして N グラム言語モデルが使われることが多い．提案分割手法でも，N グラム言語モデルによる確率に基づく分割文の指標 $Prob$ を定義し利用する．$Prob$ は N グラム言語モデルに基づく部分文の確率の総乗として式 (5.1) で定義される．この式において，P は与えられた単語列の N グラム言語モデルに基づく文としての確率を，S は分割文つまり部分文のリストをそれぞれ示し，S の各部分文に P が適用される．

$$Prob(S) = \prod_{s \in S} P(s) \tag{5.1}$$

文を特定の点で分割するか否かを判断するために，その点で分割したときとしないときの分割文の指標値を比較する．単語列の文としての確率を計算するときには，先頭と末尾の単語の出現を評価するために仮想的な語が単語列の前後に追加される．例えば “This is a medium size jacket” という文のトライグラム言語モデルに基づく確率は次のように計算される．ここで $p(\mathrm{z} \mid \mathrm{x\ y})$ は x と y が連続した出現した直後に z が出現する確率を示し，SOS と EOS は仮想的な語を示す．

$$\begin{aligned} P(\text{this is a medium size jacket}) = {} & p(\text{this} \mid \text{SOS SOS}) \times p(\text{is} \mid \text{SOS this}) \times \\ & p(\text{a} \mid \text{this is}) \times \cdots \times p(\text{jacket} \mid \text{medium size}) \times p(\text{EOS} \mid \text{size jacket}) \end{aligned}$$

$Prob$ は P の総乗であり，P は p の積からなるので，$Prob$ は p の総乗として表される．単語列 w x y z が x と y の間で分割されるとき，$Prob$ を構成する総乗の一部である $p(\mathrm{y} \mid \mathrm{w\ x}) \cdot p(\mathrm{z} \mid \mathrm{x\ y})$ は $p(\mathrm{EOS} \mid \mathrm{w\ x}) \cdot p(\mathrm{y} \mid \mathrm{SOS\ SOS}) \cdot p(\mathrm{z} \mid \mathrm{SOS\ y})$ で置き換えられることになる．

この置換により掛け合わされる p の個数は増加する．p の値は 1 以下であり，ほとんどの場合 1 未満である．このため分割点を追加すると確率に基づく指標値は小さくなる傾向がある．もしある点を分割点に加えることにより指標値が増大するならば，その点は 1 つの文の途中にあると考えるよりも，1 つの文が終了し次の文の開始する点，つまり文の分割点と考える方が尤もらしいと考えることができる．

例として “This is a medium size jacket I think it’s a good size.” という文の “jacket” と “I” の間を分割点とする場合を考える．ここでは値の比較を分かりやすく

するために p の対数値を使うことにする．この例では注目する分割点に関連する確率値は次のようになる．

(a_1)　$\log p$(I | size jacket) = -2.5244
(a_2)　$\log p$(think | jacket I) = -1.7871
(b_1)　$\log p$(EOS | size jacket) = -0.4455
(b_2)　$\log p$(I | SOS SOS) = -0.7487
(b_3)　$\log p$(think | SOS I) = -1.5927

Prob の計算式の中で分割前は (a_1)+(a_2)= -4.3115 であった部分が，分割により (b_1)+(b_2)+(b_3)= -2.7869 に置き換えられる．このため *Prob* の値は分割により大きくなり，“jacket” と “I” の間は適切な分割点と考えられる．一方，同じ文の “think” と “it” の間の位置に関連する確率は次のようになる．

(c_1)　$\log p$(it | I think) = -0.9292
(c_2)　$\log p$('s | think it) = -0.3724
(d_1)　$\log p$(EOS | I think) = -1.4225
(d_2)　$\log p$(it | SOS SOS) = -1.5504
(d_3)　$\log p$('s | SOS it) = -0.1990

この点で分割する場合，(c_1)+(c_2) = -1.3016 の部分が (d_1)+(d_2)+(d_3) = -3.1719 に置き換えられ，*Prob* の値は小さくなる．“think” と “it” の間で分割するのは不適切と判断される．

5.2.4　類似度

分割点を決めるにあたって，N グラム言語モデルは候補点周辺の数語を使った判断材料を与える．この局所的情報による判断をより大きな視点から補完するために類似度に基づく指標を導入する．ここではコーパスベース翻訳システムの学習コーパスに対する入力文の類似度を使う．コーパスベース翻訳システムはその学習コーパスにある文に類似した入力を正しく翻訳可能であると期待することができる．2 つの指標は相互補完の関係にある．N グラム言語モデルに基づく指標は文境界を考慮するが文全体は考慮していない．一方，類似度は文全体を一様に評価し文境界に重点は置いていない．

ここで 2 文間の類似度は単語列間の編集距離を用いて定義される．この編集距離は意味的要素を加味するように拡張されている．編集距離は 0 以上 1 以下の値をとるように正規化され，1 から編集距離を引いた値を類似度とする．類似度は式 (5.2) で定義される Sim_0 として示される．この式において，$|s|$ は文 s の単語数を，I と D はそれぞれ挿入，削除の個数を示す．置換は同一品詞の内容語間にのみ許される．置換のコストには Sem で示される 2 単語間のシソーラスに基づく意味距離が使われる．Sem の定義は 2.2.2 節と同じである．Sem は，式 (5.3) で示すように，K(シソーラスにおいて 2 単語の範疇が異なる概念階層数) を N (シソーラスの概念階層数) で割った値である．Sem は 0 以上 1 以下の値をとる．

$$Sim_0(s_1, s_2) = 1 - \frac{I + D + 2\sum Sem}{|s_1| + |s_2|} \tag{5.2}$$

$$Sem = \frac{K}{N} \tag{5.3}$$

例えば，式 (5.2) における s_1 と s_2 をそれぞれ，“I think it’s a good size for you” と “It’s really a good color for you” とれば，“I” と “think” がそれぞれ挿入，“really” が削除となり，“size” と “color” と置換されることになる．仮に “size” と “color” の意味距離を 0.6 とすれば，類似度 $Sim_0(s_1, s_2)$ の値は，1-(2+1+2*0.6)/(9+8) = 0.7529 となる．

コーパスに対する分割文の類似度 Sim は，式 (5.4) での Sim_1 の定義を経由して，式 (5.5) で定義される．式 (5.4) において C は与えられたコーパスつまり文の集合である．Sim_1 は文のコーパスに対する類似度，つまり与えられた文とコーパス中の文との間の類似度の最大値である．式 (5.5) において，S は分割文つまり部分文のリストである．Sim は部分文の長さで重み付けした部分文のコーパスに対する類似度の平均値である．

$$Sim_1(s) = \max\{Sim_0(s, c) | c \in C\} \tag{5.4}$$

$$Sim(S) = \frac{\sum_{s \in S} |s| \cdot Sim_1(s)}{\sum_{s \in S} |s|} \tag{5.5}$$

ALGORITHM 分割文候補生成
Input: 入力文 s
Output: 分割文のリスト
begin
　return recursive_generation(s,$\log Prob([s])$);
end

procedure recursive_generation(s,θ)
begin
　$set := \{[s]\}$;
　foreach s_1,s_2 where concatenation of s_1 and s_2 is s
　　and $\log Prob([s_1, s_2]) \geq \theta$
　　begin
　　$\delta := \log Prob([s_1, s_2])$ - θ;
　　set_1 := recursive_generation(s_1,$\log Prob([s_1])$-δ);
　　set_2 := recursive_generation(s_2,$\log Prob([s_2])$-δ);
　　foreach $S_1 \in set_1$,$S_2 \in set_2$
　　begin
　　S_{12} := concatenation of S_1 and S_2;
　　if $\log Prob(S_{12}) \geq \theta$ **then**
　　$set := set \cup \{S_{12}\}$;
　　end
　　end
　return set;
end

図 5.3. 分割文候補生成アルゴリズム

5.2.5　分割候補の生成

Sim は Sim_1 の平均値であり，Sim_1 の計算は最も類似した文をコーパスから検索することと同等である．このための検索処理はクラスタリング [13] や第 3 章で示

したグラフベースの検索手法によって効率的に実装することが可能である．しかしながら効率的な手法を使ったとしても Sim の計算には $Prob$ の計算と比べると大きなコストがかかる．そこで提案分割手法では，まず $Prob$ のみを使って分割文候補を生成する．

候補生成処理においては，まず与えられた文のみからなる分割文が候補となる．与えられた文を 2 分割した結果つまり 2 要素からなる分割文のうち $Prob$ の値が分割前に比べ小さくないものに対して，2 要素それぞれを与えられた文として再帰的に処理を繰り返す．再帰処理の結果が組み合わされて基の文の分割文候補となる．この処理を通して，基の文よりも小さな $Prob$ を持つ分割文はふるい落とされる．基の入力文のみからなる分割文は常に候補生成処理の結果に含まれる．以上のアルゴリズムを図 5.3 に示す．分割文は [と] で囲んだ表記で示す．

ここで分割文候補生成処理を例示する．候補生成処理は次の s_0 を入力文として受け取る．

s_0 = “This is a medium size jacket I think it’s a good size for you try it on please.”
$\log Prob([s_0])$ = -34.8877

$Prob([s_0])$ より小さくない $Prob$ を持つ 2 分割結果は 1 つのみ見つかる．この分割文は次の s_1，s_2 を使って $[s_1, s_2]$ と表される．

s_1 = “This is a medium size jacket”
s_2 = “I think it’s a good size for you try it on please”
$\log Prob([s_1, s_2])$ = -33.3631

この分割で確率の対数値は次の δ_0 だけ増加する．

$\delta_0 = \log Prob([s_1, s_2])$ - $\log Prob([s_0])$ = 1.5246

s_1 と s_2 に対して候補生成処理は再帰的に適用される．この再帰処理において，確率の対数値は δ_0 の範囲内で減少することが許される．再帰処理の結果は組み合わせられる．この組み合わせにより作られた分割文のうち $Prob([s_0])$ より小さくない $Prob$ を持つものが分割文候補に加えられる．s_1 からは十分に高い確率を持つ 2 分割結果は得られない．s_2 からは $[s_3, s_4]$ と $[s_5, s_6]$ が得られる．

s_3 = “I think it’s a good size for you”
s_4 = “try it on please”
s_5 = “I think it’s a good size”
s_6 = “for you try it on please”

再帰処理はさらに続けられ, 最終的に下に示す 5 つの候補が, その $Prob$ が $Prob([s_0])$ より小さくない分割文として得られる．この最終結果においては / で分割点を示し, 左の数字で $Prob$ の値による候補の順位を示す．5 番目の候補は入力文そのものである．

1. This is a medium size jacket / I think it’s a good size for you try it on please
2. This is a medium size jacket / I think it’s a good size for you / try it on please
3. This is a medium size jacket / I think it’s a good size / for you try it on please
4. This is a medium size jacket / I think it’s a good size / for you / try it on please
5. This is a medium size jacket I think it’s a good size for you try it on please

5.2.6　最良分割の選択

次に候補の中から，$Prob$ だけでなく Sim も考慮した指標に基づき，最良の分割文を選択する．選択のための指標として $Prob$ と Sim の積を使う．この指標は式 (5.6) の $Score$ として定義される．この式において，λ は 0 から 1 までの値であり Sim に重みを与える．特に λ が 0 ならば，提案分割手法は $Prob$ のみを使うことになる．また λ が 1 ならば，$Prob$ により候補が生成され，その中から Sim により選択されることになる．

$$Score = Prob^{1-\lambda} \cdot Sim^{\lambda} \tag{5.6}$$

ここで 5.2.5 節と同じ例を使って説明する．この例では特に “for you” 付近の分割に曖昧性があり複数の分割文候補が作られている．各候補について Sim さらに $Score$ が計算される．もし “this is a medium size jacket”, “I think it’s a good size for you”, “try it on please” のそれぞれに類似した文がコーパス中に存在すれば, この 2 番目の候補が最高の $Score$ を持ち最良分割文として選択されることが期待できる．

5.3 評価

5.3.1 実験条件

提案分割手法を実験を通して評価した．本手法の目的は機械翻訳の性能を上げることである．この目的に沿った評価として，いくつかの機械翻訳システムを使い，入力を分割しないときとしたときの翻訳結果を比較した．以下，実験条件について説明する．

機械翻訳システム

英日機械翻訳システムを使って分割手法の効果を評価した．ここでは提案分割手法の一般性を検証するために，2.2 節で説明した D^3 のみならず複数のシステムを使った．使用したシステムは，階層的句アラインメントに基づく翻訳器 (Hierarchical Phrase Alignment-based Translator，以下，HPAT と記す) [22]，D^3，ATR 統計翻訳器 (Statistical ATR Translator，以下，SAT と記す) [49] の 3 つのコーパスベース翻訳システムである．前 2 つは用例翻訳システム，最後の 1 つは統計翻訳システムである．

HPAT は句単位の翻訳表現を使った用例翻訳システムである．学習段階では，対訳文から 2 言語の解析木が作られ，2 つの解析木から互いに同等な句が抽出され，抽出された句の組から変換パターンが作られる．翻訳実行は変換パターンに従う．まず変換パターンの原言語側を使って原言語構造が作られる．そして原言語構造を対応する変換パターンで示された目的言語構造で置き換える．入力中のパターンに対応する部分とパターンを照合する際，HPAT は式 (5.3) の Sem と同じ意味距離を使用する．また HPAT には既に提案手法とは別の木構造解析に基づく文分割手法 [17] が組み込まれている．

SAT は用例に基づくデコーダを使った統計翻訳システムである．統計翻訳のデコーダは想定したモデル下で最大確率を持つ訳を探索する．SAT においては，対訳コーパスから検索された類似用例の訳を初期値として探索を開始する．類似度としては，情報検索でしばしば使われる TF/IDF 基準 [27] と編集距離を組み合わせて用いる．検索された訳はより良い訳を求めて書き換えられる．どのように書き換るかは貪欲探索のアプローチにより決められる．

テストセット

模擬対話コーパス (Machine-Translation-Aided bilingual Dialogues, 以下, MAD と記す) [44] 中の英語発話をテストに用いる．MAD は日本語話者と英語話者が，タイピストと機械翻訳システムからなる模擬対話翻訳システムを通して行った対話集である．この MAD から抜き出された英語発話をテストセットにする．これらは人間話者の発話であり，機械翻訳システムが生成した翻訳結果は含まない．テストセットの各発話を機械翻訳システムの入力文とする．テストセットは 505 文から成り，その平均文長は 9.52 語/文である．

学習コーパス

旅行会話基本表現集 BTEC と自然発話音声言語データベース (the bilingual travel conversation corpus of Spoken Language，以下，SLDB と記す)[44] を学習コーパスとして利用する．BTEC は旅行会話に頻出する表現を集めたものである．SLDB は日本語話者と英語話者が通訳を介して行った会話の書き起こしである．これらのコーパスでは日本語と英語が文単位で対応している．

文献 [24] は MAD タスクを扱うためには BTEC と SLDB の両者が必要であることを示している．それを受けて本実験では，MAD の発話を翻訳するために，BTEC の 152,170 対訳と SLDB の 72,365 対訳とを合わせて HPAT，D^3，SAT の学習コーパスとしている．この学習コーパスの英語部分は，提案分割手法のための言語モデルの学習や類似度の計算にも利用される．学習コーパスの統計情報を表 5.1 に示す．ここでパープレキシティは単語トライグラムを使った値である．テストセットの学習コーパスに対するパープレキシティは 63.66 である．

表 5.1. コーパスサイズ (BTEC+SLDB)

	英語	日本語
文数	224,535	
総語数	1,589,983	1,865,298
平均文長	7.08	8.31
語彙数	14,548	21,686
パープレキシティ	27.58	27.37

シソーラス

角川類語新辞典 [53] の階層構造に基づく 80,250 語のエントリを持つシソーラスを使った．シソーラスは分割手法や HPAT，D^3 において意味距離を計算する際に参照される．

分割手法の実装

言語モデルは単語トライグラムモデルとし，Good-Turing ディスカウンティングを使った．モデルの学習は CMU-Cambridge 統計的言語モデルツールキット [11] により行った．1 入力文あたりの部分文の個数は最大 4 個とした．Sim の重みである式 (5.6) 中の λ は 0, 1/2, 2/3, 3/4，1 の 5 個の値のいずれかの値をとるようにした．確率に基づく指標の方が類似度よりも値の巾が大きいため，重みは類似度側に傾けるようにしている．

評価

入力文の分割有無の条件下での翻訳文の品質を比較した．翻訳品質の評価には以下のような客観評価指標と主観評価指標を用いた．

客観指標として BLEU スコア [32]，NIST スコア [14]，mWER [45] を使った．BLEU および NIST は翻訳結果と参照訳の一致する割合を示す．mWER は翻訳結果が参照訳と異なる割合を編集距離に基づいて計算する．翻訳結果の品質が高くなれば mWER は小さくなり，BLEU スコアと NIST スコアは大きくなると考えられている．各指標ともに各テスト文につき 15 文の正解翻訳例を参照訳とした．

今回用いた主観指標は，2 つの異なる条件下での翻訳結果の一対比較法による評価を示す．各入力文毎に翻訳結果が，原言語を理解できる目的言語のネイティブにより比較され，一方の訳が他方の訳に比べ良いかどうかの判断が下される．判断は勝ち，負け，引き分けで表現される．主観指標は基準となる条件と比べたときの翻訳結果の良し悪しを示す．主観指標は，式 (5.7) で定義されるように，勝ち数と負け数の差をテスト文数で割ったものである．

$$\text{主観指標} = \frac{\text{勝ち数} - \text{負け数}}{\text{テスト文数}} \tag{5.7}$$

表 5.2. 翻訳品質: テストセット 505 文に対して分割を行った場合と行わない場合の比較

		分割無	分割有: $Score$ 別 (P は $Prob$ を，S は Sim を示す)				
			P^1S^0	$P^{1/2}S^{1/2}$	$P^{1/3}S^{2/3}$	$P^{1/4}S^{3/4}$	P^0S^1
分割入力文数		0	237	236	236	235	233
HPAT	BLEU	0.2979	0.3179	0.3201	0.3192	0.3193	0.3172
	NIST	7.1030	7.2616	7.2618	7.2709	7.2748	7.2736
	mWER	0.5828	0.5683	0.5665	0.5666	0.5658	0.5703
	主観指標		+6.9%	+8.7%	+10.1%	+10.1%	+9.5%
	勝ち数		89	95	99	99	104
	負け数		54	51	48	48	56
	引分数		94	90	89	88	73
D^3	BLEU	0.2992	0.3702	0.3704	0.3685	0.3695	0.3705
	NIST	2.1307	5.7809	5.8524	5.9115	5.9786	6.2545
	mWER	0.5844	0.5432	0.5433	0.5434	0.5424	0.5440
	主観指標		+20.6%	+21.8%	+21.8%	+22.4%	+23.0%
	勝ち数		141	145	145	146	151
	負け数		37	35	35	33	35
	引分数		59	56	56	56	47
SAT	BLEU	0.3856	0.4312	0.4330	0.4337	0.4326	0.4277
	NIST	4.1302	5.4422	5.5028	5.5704	5.5191	5.6945
	mWER	0.5510	0.5244	0.5241	0.5240	0.5235	0.5246
	主観指標		+13.3%	+13.5%	+13.5%	+14.1%	+14.5%
	勝ち数		114	115	116	117	119
	負け数		47	47	48	46	44
	引分数		76	74	72	72	70

5.3.2　入力分割による翻訳品質向上効果

表 5.2 に，HPAT，D^3，SAT の 3 システムの 6 条件下での翻訳結果の評価値を示す．条件は分割の有無，つまり入力文を分割するかどうかで分けられる．分割する場

合，入力文は $Prob$ により分割文候補に分割され，$Prob$ と Sim を使った $Score$ により選択される．この選択時の $Prob$ と Sim の重みを，$Prob$ のみを使う条件から，Sim のみを使う条件まで変化させている．主観指標の基準は分割無しの条件である．

テストセット 505 文中，選択が必要だった文，つまり複数の分割文候補が生成されたものは 237 文である．この 237 文において，候補の平均数は 5.07，最大数は 64 である．この 237 文の平均文長は 12.79 語/文である．237 文のセットの学習コーパスに対するパープレキシティは 73.87 である．

この表ははっきりとした傾向を示している．分割無しと分割有りの条件とでは評価値に明確な差がある．特に D^3 では差が大きい．分割有りの複数の条件の中での差は比較的小さくなるが，Sim を使う場合は使わない場合に比べ主観指標が，HPAT で 3.2% 増，D^3 で 2.4% 増，SAT で 1.2% 増となり着実に良くなっている．客観指標の中では NIST が主観指標との相関がよい．

表 5.3 に分割無しの条件，分割付きの中で典型的な 2 つの条件，つまり選択に $Prob$ のみ使う条件と Sim のみ使う条件での翻訳結果を例示する．上側の例では D^3 と SAT が，下側の例では HPAT と SAT が，それぞれ Sim のみ使う条件において，分割無しの場合より良い翻訳結果を出している．

一方表 5.4 は比較的長い入力文にも関わらず分割されなかった例を示す．この例では “Excuse me” は文法的に独立した文と見なすことが可能である．しかし学習コーパスの中で “Excuse me” はしばしば文の一部として現れている．この分割無しの入力に対して，D^3 において情報の一部欠落がある点を除き概ね妥当な訳が得られている．

5.3.3 類似度を用いた選択の効果

表 5.5 から分割文選択時の類似度使用による効果を確かめることができる．この表は，テストセットの中で複数候補からの選択が必要となった 237 文における評価値を示す．主観指標の基準は $Prob$ のみを使った選択である．この表の「変化数」は，$Prob$ 基準では最良となる候補以外の分割文候補の選択されたテスト文数を示す．また「理想」条件では，客観評価値から見て理想の選択となるように，翻訳結果の mWER が最良となるような候補を選択する．この理想の選択を手続き的な面から説明すると以下のような処理を行うことになる．まず各入力文に対する全ての分割文候補を各機械翻訳システムで実際に翻訳しその結果について mWER を計算する．そ

して各入力文に対する各機械翻訳システムについて得られた複数の翻訳結果の中から mWER が最良つまり最小となる結果に対応する分割文候補を選択する．このような処理による選択を理想の選択としている．理想の選択では最良の分割文はシステム毎に異なる．表中「理想」欄の「変化数」には HPAT，D^3，SAT についての値を示している．

この表から以下の傾向が読み取れる．この実験では $Prob$ と Sim を使った場合に変化数は小さい．つまり選択結果が変わる場合は少ない．$Prob$ と Sim の重みが等しい条件での変化の絶対数は小さいが，主観評価によると対応する翻訳結果はほとんどの場合に良くなり，悪くなる訳は 1 つのみである．全体として Sim の重みが大きくなるほど主観指標も良くなっている．また NIST も Sim の重みが大きいほど良い値を示している．「理想」条件では期待通り他のほとんどの条件を凌いでいるが，D^3 と SAT の主観指標と NIST の評価値では，Sim のみを使った条件より悪い値を示している．このことから D^3 と SAT の Sim のみを使った分割文選択は理想の選択に近いと言える．

ここまで NIST と主観指標の間には，他の客観指標と主観指標の間に比べ，良い相関があることが観測されてきた．NIST スコアでは他の客観指標と違い，翻訳結果と参照訳を比べる際に情報量による重み付けがなされている．分割文の翻訳結果の比較では，より情報量の多い翻訳をより良い翻訳であると人間が判断したと考えることができる．

5.3.4　類似度を用いた選択のコスト

分割文候補には $Prob$ 基準による順位を付けることができる．表 5.6 中，(A) は順位が 2 位以下の候補が選択された場合の数を，(B) は (A) において選択された候補の平均順位を，(C) は同じく (A) における最大順位を示す．

この表により選択される順位は上位に集まっていることが分かる．Sim のみを使う条件以外では，順位は常に 2 位である．このことにより選択の効果を保ちながら候補数を制限し，類似度の計算回数を抑えることが可能となる．そのため提案手法は合理的な実行時間で効果を持つ処理として実装することが可能である．

5.3.5 シソーラスを用いることによる効果

Sim の計算の際シソーラスを使わない条件で追加実験を行った．この場合，Sim の定義において全ての Sem を 0.5 と仮定する．シソーラスの使用は分割文の選択に意味的要素を加え，良い翻訳結果につながることが期待される．特に HPAT と D^3 では，Sim と同じ定義の意味距離と同じシソーラスを使っているので，特に効果が期待される．

237 文に関するシソーラスを使わない条件での評価を表 5.7 に示す．シソーラスを使った場合の表 5.5 と比べると，HPAT と D^3 では，$Score$ 中 Sim の重みが小さいとき主観指標は悪く，Sim の重みが大きいとき主観指標は良くなっている．SAT ではその逆である．しかしそれらの差は小さく，シソーラスを使うことによる明確な効果は確認されない．

5.3.6 まとめ

コーパスベース翻訳の翻訳品質向上を目的とした入力文分割手法の評価を行った．提案分割手法は N グラムに基づき分割点の候補を生成し，類似度を用いて分割点の最良の組合せを選択する．句単位の用例翻訳，文単位の用例翻訳，統計翻訳の異なる手法による 3 つのコーパスベース翻訳システムを使って評価実験を行った．実験結果は，いずれのシステムに対しても提案分割手法が有効であることを示している．類似度の使用は翻訳品質を改善し，さらに提案手法は合理的な実行時間で実現可能である．

表 5.3. 入力文分割と翻訳の例: 入力文，$Prob$ のみを使った $Score$ での分割 (P^1S^0)，Sim のみを使った $Score$ での分割 (P^0S^1)，および，それぞれを 3 システムで翻訳した結果

入力文	Here is your bill I have included a tip on your credit card slip please sign at the bottom
HPAT	請求書です。クレジットカードの伝票のチップは含んでしまいました。下の欄に署名して下さい。
D^3	はいでは請求書をお返しいたしますので下のほうに伝票お願いできますでしょうか。
SAT	チップ込みですかクレジットカードになりました伝票を下にサインをお願いします。
P^1S^0	Here is your bill / I have included / a tip on your credit card slip please sign at the bottom
HPAT	請求書です。含んでしまいました。クレジットカードの伝票のチップで下の欄に署名して下さい。
D^3	こちらが勘定書でございます。含むを負いました。それにサインするをお願いします。
SAT	こちらが勘定書でございます。私も含まれています。チップはクレジットカードの伝票番下にサインしてください。
P^0S^1	Here is your bill / I have included a tip on your credit card slip / please sign at the bottom
HPAT	請求書です。クレジットカードの伝票のチップは含んでしまいました。下の欄にサインをお願いします。
D^3	こちらが勘定書でございます。チップはカード払いに含めました。それにサインするをお願いします。
SAT	こちらが勘定書でございます。チップはカード払いに含めました。一番下にお願いします。
入力文	Well please go back to the carousel and wait a little while perhaps your bag will come soon
HPAT	戻るのですかターンテーブルに行くたぶん少し待て鞄はもうすぐ来ます。
D^3	*No output*
SAT	おそらく袋に入れてしばらく待っている台へ帰ります。
P^1S^0	Well please go back to the carousel and wait a little while / perhaps your bag will come soon
HPAT	戻るのですかターンテーブルに行く少し待って下さい。たぶん鞄だとすぐ来ます。
D^3	行ってお待ちください。たぶんすぐにお伺いいたします。
SAT	台へ帰りますのでしばらくお待ちください。たぶんカバンが参ります。
P^0S^1	Well / please go back to the carousel / and wait a little while / perhaps your bag will come soon
HPAT	えーと。ターンテーブルに帰って下さい。少し待っていただきます。たぶん鞄だとすぐ来ます。
D^3	そうですね。回転木馬へいってください。電話をおになってしばらくお待ちください。たぶんすぐにお伺いいたします。
SAT	えーと。台へ戻ってください。しばらくお待ちください。たぶんカバンが参ります。

表 5.4. 入力文と翻訳の例: 分割が不要でかつ実際に分割されなかった入力を 3 システムで翻訳した結果

input	Excuse me can you tell me where I can catch a bus from the airport to downtown
HPAT	すみません。どこで市内行きの空港からのバスに乗れますか教えてもらえませんか。
D^3	どこでバスに乗れますか教えてもらえませんか。
SAT	空港からダウンタウン行きのバスはどこで乗れますか。

表 5.5. 翻訳品質: 選択処理の実行されたテストセット 237 文に対して類似度を使った場合と使わない場合の比較

		$Score$ (P は $Prob$ を，S は Sim を示す)					理想
		P^1S^0	$P^{1/2}S^{1/2}$	$P^{1/3}S^{2/3}$	$P^{1/4}S^{3/4}$	P^0S^1	
変化数			10	19	25	91	HPAT 111 D^3 111 SAT 131
HPAT	BLEU	0.3004	0.3036	0.3022	0.3025	0.2994	0.3351
	NIST	7.1883	7.1911	7.2034	7.2068	7.1993	7.3057
	mWER	0.6363	0.6324	0.6328	0.6310	0.6405	0.5820
	主観指標		+3.4%	+3.8%	+3.8%	+5.9%	+14.8%
	勝ち数		8	12	15	40	59
	負け数		0	3	6	26	24
	引分数		2	4	4	25	28
D^3	BLEU	0.3310	0.3316	0.3291	0.3308	0.3340	0.3917
	NIST	6.0700	6.1687	6.2450	6.3372	6.6778	5.3250
	mWER	0.6181	0.6183	0.6185	0.6164	0.6197	0.5567
	主観指標		+3.4%	+3.4%	+5.5%	+6.3%	+5.5%
	勝ち数		8	10	15	37	41
	負け数		0	2	2	22	28
	引分数		2	7	8	32	42
SAT	BLEU	0.3901	0.3928	0.3913	0.3909	0.3830	0.4451
	NIST	5.5920	5.6859	5.7875	5.7098	5.9728	5.7818
	mWER	0.5990	0.5984	0.5981	0.5971	0.5995	0.5214
	主観指標		+3.0%	+3.8%	+4.6%	+7.2%	+6.8%
	勝ち数		8	12	14	41	48
	負け数		1	3	3	24	32
	引分数		1	4	8	26	51

表 5.6. 選択された分割文候補の $Prob$ に基づく順位 (P は $Prob$ を，S は Sim を示す)

$Score$	$P^{1/2}S^{1/2}$	$P^{1/3}S^{2/3}$	$P^{1/4}S^{3/4}$	$P^{0}S^{1}$
(A) 1 位以外が選ばれた場合の数	10	19	25	91
(B) (A) の場合の平均順位	2.00	2.00	2.00	4.01
(C) (A) の場合の順位の最大値	2	2	2	20

表 5.7. 翻訳品質: シソーラスを使わない条件下，選択処理の実行されたテストセット 237 文に対して類似度を使った場合と使わない場合の比較

		$Score$ (P は $Prob$ を，S は Sim を示す)					理想
		P^1S^0	$P^{1/2}S^{1/2}$	$P^{1/3}S^{2/3}$	$P^{1/4}S^{3/4}$	P^0S^1	
変化数			10	19	26	93	HPAT 111 D^3 111 SAT 131
HPAT	BLEU	0.3004	0.3027	0.3034	0.3039	0.2973	0.3351
	NIST	7.1883	7.1830	7.1921	7.2003	7.1741	7.3057
	mWER	0.6363	0.6342	0.6320	0.6321	0.6346	0.5820
	主観指標		+1.7%	+3.8%	+3.4%	+6.3%	+14.8%
	勝ち数		6	13	15	40	59
	負け数		2	4	7	25	24
	引分数		2	2	4	28	28
D^3	BLEU	0.3310	0.3301	0.3310	0.3290	0.3370	0.3917
	NIST	6.0700	6.1387	6.2414	6.3341	6.6739	5.3250
	mWER	0.6181	0.6196	0.6188	0.6198	0.6175	0.5567
	主観指標		+3.0%	+4.6%	+5.9%	+7.6%	+5.5%
	勝ち数		7	12	16	41	41
	負け数		0	1	2	23	28
	引分数		3	6	8	29	42
SAT	BLEU	0.3901	0.3919	0.3929	0.3896	0.3758	0.4451
	NIST	5.5920	5.7101	5.7867	5.8006	5.9118	5.7818
	mWER	0.5990	0.5990	0.5980	0.5979	0.6025	0.5214
	主観指標		+3.0%	+5.1%	+4.2%	+5.5%	+6.8%
	勝ち数		7	13	14	40	48
	負け数		0	1	4	27	32
	引分数		3	5	8	26	51

第 6 章

後処理による翻訳誤り対策

機械翻訳システムの性能向上の目的のためには，翻訳エンジンの工夫だけでなく，後処理によって結果をより適切な訳に変換するアプローチもある．このアプローチの長所として，様々な翻訳エンジンに適用可能である点が上げられる．ここでは，コーパスベース翻訳において特徴的に現れる誤りへの対応を試みる．文単位の用例翻訳システムの翻訳誤りに関する 4.5 節の分析では，入力文に含まれない情報が翻訳結果に現れるタイプの誤りが相対的に多いことが分かった．この誤りは，このシステムに限らず，多くのコーパスベース翻訳システムで観察される．本章では，この誤りを後処理により修正する手法を提案し評価する．

6.1 コーパスベース翻訳における特徴的な誤り

近年，コーパスベースの翻訳技術の研究開発が盛んになってきており，その性能も向上している [55]．しかしながら用例翻訳や統計翻訳などのコーパスベース翻訳では，翻訳知識を人間の手を介さず自動的に構築するゆえに，ときとして人間にとって考えられないような翻訳誤りを犯す．特に，原文で全く触れてもいない概念を表す語が訳文に現れる場合，その誤りの奇異な印象は強く，例え全体的な性能が高くても，システムへの不信感をユーザに与えてしまう．その一例として 4.5 節で取り上げた翻訳誤りを下に示す．この例では，入力文とは関連のない “departure” という語が訳文に現れてしまっている．

入力文: お名前とお部屋番号を教えて下さい

翻訳結果:　What is the name and departure room number?

この例のように入力文と関連なく訳文中に現れた語を，我々は湧き出し語と名付ける．

本章では，湧き出し語問題に対処するために，後編集により翻訳誤りを自動修正するアプローチについて論ずる．翻訳システム内部の改良ではなく，後編集のアプローチをとることで，複数種類の翻訳システムの翻訳結果への適用が可能となる．一般に後編集による修正には削除・挿入・置換の操作があるが，今回は湧き出し語の削除に焦点を当てる．提案手法では，統計翻訳で用いられる単語翻訳モデルを利用して削除すべき訳語の候補を検出し，対訳コーパスから得られた用例を使って候補を絞り込む．

以降の節では，湧き出し語の発生の仕組み，我々のアプローチと関連研究，提案手法と実験について報告する．実験では，複数の翻訳システムを対象とし，訳語自動削除による翻訳精度向上効果を検証する．

6.2　湧き出し語問題

どのような方式の機械翻訳システムでも，完全なものはなく，何らかの翻訳誤りを犯す．対訳コーパスから翻訳知識を学習するコーパスベース翻訳方式のシステムでも，個々に様々な誤りを犯すが，共通して目に付くのは湧き出し語の問題である．例えば国際ワークショップ IWSLT-2004 評価キャンペーン [1, 55] ではコーパスベース翻訳システムは高い翻訳精度を示したが，やはり湧き出し語問題が観察される．特に好成績を上げた統計翻訳システムにおいて残された問題の中で大きなものが湧き出し語である[*1]．

湧き出し語の概念は，ユーザに提示される翻訳結果で観察される問題を示し，それがいかに生成されたかのシステムの内部事情を区別しない．しかしコーパスベース翻訳という枠内で考えると，湧き出し語の発生原因の特徴をとらえることができる．湧き出し語の多くは単語アラインメント [27] に起因する．単語アラインメントとは，対訳を構成する原文と訳文の単語間の対応関係を求める処理およびその結果の情報を指す．単語アラインメントは，コーパスベース翻訳において翻訳知識の学習に必須の基本的な処理および情報と位置づけられる．

[*1] IWSLT-2004 で使われた，学習用，開発用，テスト用のコーパス，参照訳，各システムの翻訳結果のすべてが GSK(言語資源協会，http://www.gsk.or.jp/) を通じて一般に公開される．

6.2.1 用例翻訳における問題

まず用例翻訳方式における湧き出し語について例を上げて説明する．以下，6.1 節で取り上げた例について，その発生過程を示す．

入力文: お名前と お部屋番号 を教えて下さい

に対して，次の類似用例が見つかる．

原文: 便名と時間を教えて下さい
訳文: What is the flight number and departure time?

入力文は，用例原文の「便名」を「お名前」,「時間」を「お部屋番号」でそれぞれ置換したものとみなされる．ここでシステムは,「便名」と「時間」の用例訳文中の対応箇所はそれぞれ “flight number” と “time” であると判断する．これらの用例訳文中の対応箇所に「お名前」と「お部屋番号」の訳語が入れられ，次の翻訳結果が得られる．

翻訳結果: What is the name and departure room number?

結果として “departure” が残り，湧き出し語となってしまう．この原因は,「時間」の対応箇所が “time” と判断され “departure” が含まれなかったこと，つまり不適切な単語アラインメントにある．ここでは用例翻訳の一方式の例を使って説明したが，単語アラインメントの誤りに起因する湧き出し語は他の用例翻訳方式でも発生する．

6.2.2 統計翻訳における問題

一方，統計翻訳においては，多くの対訳表現から得られる特徴が統計的に処理されるので，それぞれの対訳表現は，用例翻訳の例のように端的な動作を引き起こすのではなく，翻訳結果に見られるいくつかの傾向を強化する．ここでは単語アラインメント結果から翻訳モデルを学習する際，前節の例で類似用例として示した対訳表現が学習データに存在する場合を考える．この対訳表現において，もし単語アラインメント中で “departure” がいずれの原言語単語にも対応していないとすると，冠詞などと同様に原言語単語との明確な対応がなくても出現する語として “departure” が扱わ

れる確率が大きくなる．そのため，“departure” が誤って訳文に挿入されてしまう可能性が出てくる．逆に「時間」の対応箇所が “departure time” であると判断された場合，「時間」と “departure” との対応確率が大きくなる．そのため “departure” とは関連のない「時間」を含んだ入力文に対する訳文に “departure” が誤って出現する可能性が出てくる．もちろん翻訳モデルは，他の多くの対訳表現から生成されるのでこれらの誤りは淘汰されることが期待される．しかし統計翻訳には，モデルではとらえ難い現象がある（モデルの問題），高度なモデルでは最適なパラメータの発見が保証されない（学習の問題），翻訳時の探索空間が大きく最適な訳文を得るのが難しい（探索の問題）といった問題が存在する [29]．このため翻訳結果に誤りが残ってしまうことがあり得る．統計翻訳における湧き出し語の具体例は，6.5.4 節において実験結果を考察する際に改めて取り上げる．

6.3　問題へのアプローチ

6.3.1　後編集による修正

湧き出し語問題への対処として，個々の翻訳システム自体の改良，特にその利用する単語アラインメントの改良が考えられる．しかし本章において我々は，機械翻訳結果を後編集により自動修正するアプローチを取り上げる．

後編集アプローチには，特定の翻訳システムに限らず複数のシステムに適用可能という利点がある．複数の翻訳システムの利用例としてセレクタ・アーキテクチャ [2] が上げられる．セレクタ・アーキテクチャでは，複数の翻訳システムの出力した複数の訳文を統計的に評価し，スコアが最良の訳を選択する．これにより，性質の異なる翻訳システムが互いに補い合い，より多くの原文に対して良い訳文を得ることが可能となる．この枠組においては，個々の翻訳システムは固定ではなく他のシステムへの差し替えも考えられ，特定の翻訳システムに対象を限定しない後編集アプローチが有効となる．

6.3.2　関連研究

湧き出し語問題および後編集アプローチに関連する先行研究として，文献 [57] の翻訳自動校正，文献 [7] の翻訳文の誤り検出，および，文献 [18] の統計翻訳の貪欲デコーディングの各研究が上げられる．

文献 [57] では，我々と同様に後編集により訳文を修正するが，入力文の内容をいかに正確に伝えるかという問題ではなく，いかに自然な文を生成するかという問題に重点を置く．その提案手法では，翻訳結果とそれを人手で校正した結果を使って校正規則を学習し，その規則に従って翻訳文を修正する．それに対して我々は，入力文と翻訳結果の対応関係に基づいて翻訳文を修正する．つまり入力文の内容を正確に伝える問題に重点を置く．両者のアプローチは互いに補完関係にあると考えられる．

文献 [7] では，単語レベルでの翻訳文の誤り検出を行う．この研究では統計翻訳システムを対象とし，翻訳結果中の各単語について，いくつかの指標を用いて正誤判定を行う．この手法では対訳コーパスと対象翻訳システムを使った学習を行う．すなわち，対訳コーパスの原文を翻訳しNベスト訳を出し，対訳コーパスの参照訳と比べることにより単語の正誤を判定し，この判定を基準として各指標値に対する単語の正誤分類をナイーブベイズ法で学習する．実験ではNベストのNを 1000 として学習および評価を行っている．実験結果として，一番精度の良かった指標は，Nベスト順位で重み付けした単語の出現頻度（ランク重み付け頻度）であり，僅差で IBM モデル 1 [8] の単語翻訳モデルに基づく指標が続く．

ランク重み付け頻度を使うには，翻訳システムの出すNベスト訳が必要である．ランク重み付け頻度の出す信頼度は，翻訳システムの判断，つまり翻訳システムの作成した訳とその順位に依存する．一方，単語翻訳モデルは，特定の翻訳システムに依存せず，それを利用するためには，正誤判定実行時のNベスト訳やNベスト訳を使った学習は必須ではない．我々は，Nベスト訳や学習を利用せず，単語翻訳モデルを使った誤り語検出処理とその拡張を提案する．

文献 [18] では，最適な翻訳文を探索する統計翻訳手法が提案されている．この手法は，翻訳結果の後編集による修正ではないが，より良い翻訳文を探索するために訳文候補に変形操作を加える．この手法では，5 種類の変形操作を定義し，適当に与えられた翻訳文の元となる単語列に対して，評価関数値を良くする操作がある限り，それを繰り返し適用する．ここで定義された変形操作はより一般的なものであるのに対し，我々の提案の要点は，単語翻訳モデルによって検出される誤りに修正箇所を絞ることにある．また提案手法は，翻訳システムが結果として出力した文，つまりシステムが最良と判断した翻訳文を対象とする．

6.4　単語翻訳モデルを用いた訳語削除手法

機械翻訳の後編集処理として湧き出し語を削除する手法を提案する．本手法では，まず単語翻訳モデルによって削除語候補を検出し，次に対訳用例を使って候補を絞り込み，残った候補を削除する．

6.4.1　単語翻訳モデル

湧き出し語は，入力文中の単語との対応確率の小さな訳語と考えるのは自然である．この確率計算のために，提案手法では，統計翻訳モデルの原始要素である単語対応確率を利用する．

翻訳モデルは，ある言語の文 F が与えられたとき，別の言語のある文 E に翻訳される確率 $P(E|F)$ を与えるモデルである．統計翻訳でよく使われる翻訳モデルは IBM モデル [8] であり，IBM モデル 1 から 5 までのバリエーションがある．IBM モデル 1 は，原言語と目的言語の単語間の翻訳確率を示す単語翻訳モデルのみから構成される．このモデルは対訳コーパス中の対訳文における両言語の単語の共起関係から求められる．一方 IBM モデル 2 から 5 では，原言語文のある位置に現れた語が他言語の文の特定の位置に対応する確率，原言語のある単語が他言語の特定個数の単語に翻訳される確率なども考慮し組み合わせたモデルである．

今回の問題となる湧き出し語は，原文で全く触れてもいない概念を表す語が訳文に現れることにより奇異な印象を与える問題である．我々は，湧き出し語は原文中の単語の集まりから想起されない訳語であり，単語の出現位置や対応語数に関係ないととらえ，単純な IBM モデル 1 の単語翻訳モデルを利用する．

また統計翻訳では，与えられた原文 F に対して翻訳確率 $P(E|F)$ を最大にする訳文 E を求める問題を，ベイズの法則を利用して $P(F|E)P(E)$ を最大にする E を求める問題に置き換えることがよく行われる．ここで $P(F|E)$ はもとの翻訳方向とは逆方向の翻訳確率であり， $P(E)$ は言語モデル確率である．この変換により言語モデルを利用することが可能となる．しかし湧き出し語問題は 2 言語間の問題であり，我々は，訳文単独で評価する言語モデルは使わず，順方向の翻訳モデル $P(E|F)$ を用いる．

6.4.2 単語翻訳モデルによる削除語候補検出

対訳コーパスから学習した IBM モデル 1 の単語翻訳確率を利用し，翻訳文中に現れた各単語 e について，入力文と単語翻訳確率 p から見た出現数の期待値 $C(e)$ を求める．

$$C(e) = \sum_{j=0}^{m} p(e|f_j) \tag{6.1}$$

ここで，入力文は $f_1, \ldots, f_m$ の単語列であり，f_0 は NULL 単語を意味する．つまり $p(e|f_0)$ は，訳語 e が原文のいずれの語とも関連なく訳文に現れる確率を示す．$C(e)$ が十分小さな値，つまり次式のようにある小さな定数 δ 以下となる訳語 e を削除語候補とする．

$$C(e) \leq \delta \tag{6.2}$$

この枠組において NULL 単語を使うことは，原言語の単語と明確に対応しない機能語などの訳語にある程度の大きさの確率を与え，その削除を防ぐことにつながる．

単語翻訳モデルは 2 言語間の単語の対応関係を確率として与える．一方，人手で編集された対訳辞書は信頼度の高い対応関係を与えると考えられる．今日では多くの対訳辞書が電子的に利用可能となっている．人手で編集された対訳辞書を利用する場合，入力文中の原言語単語に関して辞書に示された訳語は，原言語単語からの高い翻訳確率を持つとみなして削除の対象としないこととする．

6.4.3 対訳用例による削除語の制限

単語対応確率が大きくなくても，入力文全体に対応して正しいと考えられる訳語もあり得る．この場合，単語対応確率のみで判断すると誤り語として削除される危険がある．これを防ぐために類似用例を利用する．

以下，例で示す．

入力文: もう一度名前を探してください
翻訳結果: Could you check my name again please?

これは良い翻訳だが,「探す」をはじめとする入力文中の語と “check” の翻訳確率は必ずしも高くない．例えば「探す」は “look for” や “find” などの語句に対応する場合が多く “check” への翻訳確率は大きくない．このため “check” が削除語候補となる．しかし，次の対訳用例が対訳コーパスに存在することにより “check” が正しい訳語だと判断され，削除語候補から外される．

原文:　もう一度探して下さい
訳文:　Could you check again?

ここで改めて，用例を利用して削除語を制限する判断基準を定義する．まず対訳コーパス中の対訳文を用例とし，入力文と編集距離の近い原文を持つ用例を類似用例とする．2 文間の編集距離 $dist(s_1, s_2)$ は，次の定義のように定義する．ここで，s_1 と s_2 は対象となる 2 つの文であり，$|s_1|$ と $|s_2|$ はそれぞれの単語数，I と D は，それぞれ 2 単語列の比較における挿入語数と削除語数を示す．置換は挿入と削除の組み合わせに相当し，式には現れない．

$$dist(s_1, s_2) = \frac{I + D}{|s_1| + |s_2|} \tag{6.3}$$

この距離定義に基づくコーパスからの類似用例検索は，クラスタリング [13] や第 3 章で示したグラフベースの検索手法によって効率的に行うことが可能である．

また注目する訳語 e と各類似用例について，IN を入力文に現れる用例原文の単語のリスト，OUT を入力文に現れない用例原文の単語のリストとする．先の例では，IN は { もう, 一度, 探し, て下さい }，OUT は空リストとなる．ここで入力文を ” 名前を探してください” とすると，IN は { 探し, て下さい }，OUT は { もう, 一度 } となる．

以上の準備のもと，次の 3 条件を全て満たす類似用例が存在する単語 e は削除語とはしない，と判断する．

- 用例訳文が e を含む.
- 対訳辞書を利用する場合，辞書に示された OUT 中の語の訳語に e が現れない
- 式 (6.4) を満たす.

$$\sum_{f \in OUT} p(e|f) \leq \sum_{f \in IN} p(e|f) \tag{6.4}$$

これらの条件の意図するところは，用例原文のうち入力文に使われている語句に対応する訳語は削除しない，ということである．

6.4.4 削除の実施

削除語候補となった訳語のうち，用例による制限にかからなかった語を削除語とし，訳文から削除する．

6.5 評価

複数の翻訳システムの翻訳結果に対して提案手法を適用し，その有効性を確認した．

6.5.1 実験条件

実験では，IWSLT-2004 評価キャンペーンにおける言語資源と参加した翻訳システムの翻訳結果を利用した．2.2 節で挙げた D^3 のみならず複数のシステムの翻訳結果を利用することにより提案手法の一般性を検証した．以降の実験結果の説明の中で示す ATR-H システム [41] が D^3 を使ったシステムである．ATR-H システムでは，D^3 の翻訳結果と 5.3.1 節で挙げた HPAT の翻訳結果から統計的指標に従って良い方の訳を選択する [2].

IWSLT-2004 評価キャンペーンは，旅行会話に関するコーパス BTEC を用い，参加システムの翻訳結果を評価するものである．本章の実験では，日英翻訳の supplied トラック，中英翻訳の supplied トラック，日英翻訳の unrestricted トラックを対象とした．supplied トラックでは，翻訳対象言語対に関する 2 万文の原文とその訳語からなる対訳コーパス (表 6.1) を学習セットとして与えられ，参加システムは，それ以外の言語資源を使うことは許されない．unrestricted トラックでは言語資源の制限はなく，追加の対訳コーパス，辞書，構文解析知識などの使用が許される．従って unrestricted トラックの参加システムは supplied トラックの参加システムよりも高品質であることが期待できる．各トラックとも 500 文からなるテストセットを翻訳する．テストセットの平均文長は，日英では 8.7 語/文，中英では 7.6 語/文である．

実験では，参加各システムの翻訳結果に訳語削除手法を適用した．supplied トラックを使った実験では，キャンペーンで与えられた 2 万対訳の学習セットを使って，削除語検出に用いる IBM モデル 1 を構築した．日英翻訳の unrestricted トラックに

表 6.1. コーパスサイズ: supplied トラック用

	日英		中英	
	日本語	英語	中国語	英語
文数	20,000		20,000	
総語数	209,012	188,712	182,904	188,935
平均文長	10.45	9.44	9.15	9.45
語彙数	9,277	8,074	7,643	8,191

表 6.2. コーパスサイズ: 日英 unrestricted トラック用

	日本語	英語
文数	224,535	
総語数	1,865,298	1,589,983
平均文長	8.31	7.08
語彙数	21,686	14,548

ついては，当トラックに参加した ATR-H システムの使ったコーパスを用いて IBM モデル 1 を構築し，また同システムの使った日英対訳辞書も削除語検出で利用した．このコーパスの対訳数は約 22 万（表 6.2），辞書の見出し語数は約 9 万である．

IBM モデル 1 の構築には GIZA++[31] を用いた．削除語候補検出処理においては $C(e)$ が 0.01 以下となる訳語 e を候補とした．用例による削除語制限処理においても，削除語候補検出処理と同じコーパスを使った．ここでは入力文に対する距離が 0.4 以下の原文を持つ用例を類似用例とした．類似用例が多数の場合，距離の小さい方から 100 位までの用例を使った．

6.5.2　翻訳品質への効果

各システムの翻訳結果に訳語削除手法を適用し，適用前後の翻訳自動評価値を比較した．翻訳自動評価指標として mWER [45] と BLEU スコア [32] を使用した．mWER は翻訳結果が参照訳と異なる割合を編集距離に基づいて計算する．BLEU は翻訳結果と参照訳の N グラムの一致する割合を示す．翻訳結果の品質が高くなれば mWER は小さくなり，BLEU スコアは大きくなると考えられている．mWER,

BLEU ともに各テスト文につき 16 文の正解翻訳例を参照訳とした．

日英 supplied トラックの評価結果を表 6.3 に，中英 supplied トラックを表 6.4，日英 unrestricted トラックを表 6.5 に，それぞれ示す．表中，各システムの左肩に付いている記号は翻訳方式を示す．s は統計翻訳，e は用例翻訳，h は統計翻訳と用例翻訳のハイブリッド，r はルールベース翻訳である．表は各システムについて，訳語削除を行っていないベースとなる翻訳文と削除後の翻訳文の自動評価値を示す．削除に関しては 2 種類の結果を示している．「削除」では，対訳用例による制限を行わず，削除語候補検出処理により候補となった語を全て削除している．「削除 (制限付)」では対訳用例により削除語候補を制限している．また削除率の欄は，翻訳語全体に対する削除された語の割合，訳文数 500 に対する 1 語でも削除された文の割合を示す．

結果として，日英 supplied トラックの ATR-S システムの BLEU 値を除く全指標で，評価値の差の大小はあるにせよ，削除語制限を行った場合と行わなかった場合のいずれでも，削除後の翻訳文の方がより良い評価値を得ている．この結果は，単語対応モデルを基準に翻訳誤りを修正するアプローチの有効性を示している．

一方，削除語制限を行った場合と行わなかった場合とでは，自動評価値の差は小さく，いずれの場合により良い値を示すかを言うことはできない．これは削除率の差が小さいためと考えられる．特に supplied トラックでは，unrestricted トラックに比べ削除率の変化は小さい．これは，supplied トラックでは小さなコーパスを使うため，削除語制限条件に合う類似用例が少ないことによる．つまり，この実験では削除語制限の有効性は示されていない．

6.5.3 翻訳妥当性との関連

訳語削除が有効に機能している場合，何らかの訳語削除がなされたベースとなる訳文は妥当な訳ではないことが期待される．ここでは訳語削除の有効性を，翻訳妥当性に関する主観評価結果を使って検証する．

IWSLT-2004 評価キャンペーンにおいて，各翻訳システムの翻訳結果の妥当性について主観評価がなされている．それぞれの訳文について，原言語を理解できる目的言語のネイティブ 3 名により 5 段階の評価が与えられている．評価は 5 から 1 までの数値で与えられ，大きい数字ほど妥当な訳となる．各訳文について 3 名の付けた評価の中間値が，その訳文の翻訳妥当性ランクとされ，テストセット 500 文の平均値が各システムの翻訳妥当性スコアとされる．

表 6.3. 翻訳品質: 日英 supplied トラック

システム		削除率 (%) 語	文	mWER	BLEU
	ベース	-	-	0.4196	0.4515
^{s}RWTH	削除	2.4	14.2	0.4155	0.4566
	削除 (制限付)	2.4	14.2	0.4155	0.4566
	ベース	-	-	0.4844	0.4008
^{s}ISI	削除	1.3	6.4	0.4791	0.4061
	削除 (制限付)	1.3	6.2	0.4791	0.4064
	ベース	-	-	0.5289	0.3649
^{s}IBM	削除	4.3	18.8	0.5171	0.3852
	削除 (制限付)	4.2	18.6	0.5168	0.3858
	ベース	-	-	0.6145	0.3645
^{s}ATR-S	削除	15.4	47.6	0.6047	0.3625
	削除 (制限付)	15.4	47.6	0.6047	0.3625

表 6.6，表 6.7，表 6.8 は，それぞれ日英 supplied トラック，中英 supplied トラック，日英 unrestricted トラックに関して，訳語削除の行われた文と翻訳妥当性の関係を示す．各表のベースの欄は各システムについて，空出力を除く翻訳結果のあった文数，翻訳妥当性ランク毎の文数，および，翻訳妥当性ランクの平均値を示す．削除の欄は，1 語でも訳語削除がなされた文についてのみ，ベースから翻訳妥当性ランクを抽出したものである．つまり，いずれの欄でも，個々の文の翻訳妥当性ランクとしては，削除を行わない元の訳文のランクを使い，欄毎に対象となる訳文の集合が異なっている．削除に関しては対訳用例による制限の有無により 2 種類の結果を示している．

結果として，削除のあった訳文での翻訳妥当性ランクの平均値は，ベースに比べ，削除語制限を行った場合と行わなかった場合のいずれでも，明確に悪い値になっている．その差は，日英 unrestricted トラックの ATR-H システムなど，ベースの平均ランクの良いシステムで特に大きくなっている．このことから削除の有無を基の翻訳妥当性の信頼度とみなすことができる．場合によっては，削除のある訳文は妥当な訳ではないとして棄却するという提案手法の利用法も考えられる．この利用法は，6.3.1

節で触れたセレクタ・アーキテクチャにおいて，削除のある訳と削除のない訳が混在している場合に有望である．

一方，削除語制限を行った場合は，行わなかった場合に比べ，翻訳妥当性ランクの平均値は確実に悪くなっている．supplied トラックでは，削除語制限の有無による削除語数の差異は小さく，全く差のない場合もある．しかし差のある場合は削除語制限を行った場合の方が悪くなっている．つまり削除語制限を使うことにより，より妥当性の低い翻訳文を絞り込むことができている．

6.5.4 統計翻訳への効果

当手法は IBM モデル 1 に基づく単語翻訳モデルを利用して誤り語を検出する．統計翻訳システムでは同様の翻訳モデルが利用されると考えられるが，その翻訳結果に対しても効果が確認された．つまり評価対象の大半は統計翻訳システムであったが，それらのシステムにおいて翻訳品質への効果，翻訳妥当性との関連が認められた．

最近の統計翻訳では，句レベルの翻訳モデルの有効性が認められ，それを利用する方式が主流となっている．今回対象とした統計翻訳システムの全て [41, 26, 6, 16, 47, 4, 19] が句レベルの翻訳情報を扱い，そのうち一つを除くシステムが句翻訳モデルを使っている．また多くのシステムでは，IBM モデル 1 に基づく単語翻訳モデルを，学習の初期データとして，あるいは訳文の適切性を示す指標の一つとして利用している．これらのシステムに対して提案手法が効果を持つことは，高度なモデルを利用する統計翻訳においても，単語翻訳モデルのさらなる有効利用法があることを示している．より具体的には，ある単語を訳文中で使用するかどうかの判断において，単語翻訳モデルに基づく確率値に関する条件を，必要条件として使うことの有効性を示唆する．

以下，実験結果から湧き出し語の削除例を示す．これは日英 supplied トラックの IBM システムの翻訳結果で見られた例である．このシステムは句翻訳モデルを使い，また IBM モデル 1 による単語翻訳モデルを学習の初期データとして利用している．

入力文: 航空券を家に忘れてしまいました
翻訳結果: i have a [return] ticket i left my [glasses] in the house [*2]

[*2] 英語の正書法に従わず大文字や句読点を使っていないのは，訳文の評価が lower-case only，no punctuation marks という条件下で行われたためである．

この翻訳結果にはいくつかの問題があるが，不要と思われる語句に下線を引いている．また提案手法によって削除された語を [ ] で囲んでいる．特に入力文から見て唐突な湧き出し語と見られる “return” と “glasses” が提案手法によって削除されている．

学習セットにおいて，上記の入力文中の語と湧き出し語が共起するいくつかの対訳が観察された．そのうち二つの対訳を下に示す．

原文 1:　今朝十時にボストンを出た電車にメガネを忘れました
訳文 1:　i left my glasses on the train that left boston at ten this morning

原文 2:　帰りの航空券を見せてください”
訳文 2:　your return ticket please

それぞれの対訳において網がけした部分の間でなんらかの対応がとられ，それが誤りの発生に影響を与えたと推測することができる．単語翻訳モデルでは，入力文中の各語と”return” , “glasses” との対応確率は十分小さくなり，これらの語を誤りであると判断することができた．

6.5.5　削除例

いくつかの日英翻訳実験システム [2, 49, 51, 23, 15] の使用者から湧き出し語を含んだ翻訳結果として報告された例について，訳語削除処理を適用した結果を 表 6.9 に示す．湧き出し語と他の誤りとで区別し難い部分もあるが，下線を引いた部分が入力文と関連のない余分な語句と考えることができる．これらの翻訳文に対する訳語削除処理には 6.5.1 節の日英 unrestricted トラックに対する設定を使った．[ ] で囲んだ部分が実際に削除された語である．これらの例からは，削除による誤り訂正が有効に働いていることが見て取れる．

一方，表 6.9 の例からは今後の課題も浮かび上がる．例えば (1) や (6) では，削除語に付随する不要な “the” が消されずに残ってしまう．この問題への対処として，浅い構文解析 [37] などの手法により句を括り出し，削除語に付随する語も共に削除するなどの削除範囲を調整する方法が考えられる．

また，例えば (19) では，訳文を構成する主要な語が削除され，結果として意味のない文になっている．このような訳を適切な訳になるように修正するには，湧き出し

語の削除だけでなく，適切な語への置換や訳文に不足する語の挿入などの操作が必要になる．一方で，訳語削除の結果として意味のない文となるような翻訳結果は 6.5.3 節で示唆した削除語のある訳文は棄却するというアプローチが有効な例と言える．

6.5.6 まとめ

コーパスベース翻訳の典型的な誤りである湧き出し語の問題に対処するための後編集により翻訳結果を修正する手法を評価した．翻訳自動評価値および翻訳妥当性ランクに関するデータからは提案訳語削除手法の有効性が示された．一方，訳語削除の実例からは，訳語削除手法が湧き出し語に対する誤り訂正として有効に働いていることが窺える．

表 6.4. 翻訳品質: 中英 supplied トラック

システム		削除率 (%) 語	削除率 (%) 文	mWER	BLEU
	ベース	-	-	0.4548	0.4093
^{s}RWTH	削除	2.4	12.0	0.4527	0.4139
	削除 (制限付)	2.4	11.8	0.4527	0.4138
	ベース	-	-	0.4702	0.4535
^{s}ATR-S	削除	8.4	35.6	0.4599	0.4934
	削除 (制限付)	8.3	35.6	0.4595	0.4936
	ベース	-	-	0.4716	0.4152
^{s}ISL-S	削除	3.7	20.4	0.4630	0.4288
	削除 (制限付)	3.7	20.4	0.4630	0.4288
	ベース	-	-	0.4872	0.3754
^{s}ISI	削除	3.0	12.2	0.4862	0.3819
	削除 (制限付)	3.0	12.0	0.4860	0.3818
	ベース	-	-	0.5083	0.3489
^{s}IRST	削除	12.3	46.0	0.4992	0.3984
	削除 (制限付)	12.3	46.0	0.4992	0.3984
	ベース	-	-	0.5330	0.3382
^{h}IAI	削除	7.6	39.0	0.5078	0.3602
	削除 (制限付)	7.6	39.0	0.5078	0.3602
	ベース	-	-	0.5391	0.3465
^{s}IBM	削除	4.3	16.8	0.5334	0.3567
	削除 (制限付)	4.2	16.6	0.5342	0.3561
	ベース	-	-	0.5564	0.2786
^{s}TALP	削除	5.0	23.0	0.5486	0.2877
	削除 (制限付)	4.8	22.4	0.5493	0.2875
	ベース	-	-	0.6172	0.2089
^{e}HIT	削除	12.2	42.2	0.6036	0.2361
	削除 (制限付)	12.0	41.4	0.6043	0.2361

表 6.5. 翻訳品質: 日英 unrestricted トラック

システム		削除率 (%) 語	削除率 (%) 文	mWER	BLEU
^{h}ATR-H	ベース	-	-	0.2631	0.6306
	削除	2.3	10.6	0.2608	0.6490
	削除 (制限付)	2.0	8.4	0.2596	0.6481
^{s}RWTH	ベース	-	-	0.3064	0.6180
	削除	2.4	12.8	0.2993	0.6308
	削除 (制限付)	2.1	11.4	0.3009	0.6313
^{e}UTokyo	ベース	-	-	0.4852	0.3963
	削除	9.4	33.4	0.4628	0.4484
	削除 (制限付)	8.4	29.2	0.4629	0.4464
^{r}CLIPS	ベース	-	-	0.7304	0.1320
	削除	12.4	53.6	0.6991	0.1529
	削除 (制限付)	12.1	52.6	0.6997	0.1524

表 6.6. 翻訳妥当性: 日英 supplied トラック

システム		文数	翻訳妥当性ランク別文数					翻訳妥当性
			5	4	3	2	1	ランクの平均
	ベース	500	181	90	68	76	85	3.412
^{s}RWTH	削除	71	8	8	12	19	24	2.394
	削除 (制限付)	71	8	8	12	19	24	2.394
	ベース	499	157	74	53	88	127	3.092
^{s}ISI	削除	32	6	0	2	11	13	2.219
	削除 (制限付)	31	5	0	2	11	13	2.129
	ベース	499	130	77	71	103	118	2.996
^{s}IBM	削除	94	5	10	18	26	35	2.191
	削除 (制限付)	93	4	10	18	26	35	2.191
	ベース	473	66	29	34	79	265	2.053
^{s}ATR-S	削除	238	9	4	13	40	172	1.479
	削除 (制限付)	238	9	4	13	40	172	1.479

表 6.7. 翻訳妥当性: 中英 supplied トラック

システム		文数	翻訳妥当性ランク別文数					翻訳妥当性
			5	4	3	2	1	ランクの平均
	ベース	500	174	86	66	83	91	3.338
^{s}RWTH	削除	60	12	5	9	14	20	2.583
	削除 (制限付)	59	11	5	9	14	20	2.542
	ベース	495	143	69	54	93	136	2.980
^{s}ATR-S	削除	178	21	19	26	38	74	2.298
	削除 (制限付)	178	21	19	26	38	74	2.298
	ベース	496	149	79	50	95	123	3.073
^{s}ISL-S	削除	102	14	14	11	26	37	2.431
	削除 (制限付)	102	14	14	11	26	37	2.431
	ベース	499	142	89	64	80	124	3.090
^{s}ISI	削除	61	7	10	11	13	20	2.525
	削除 (制限付)	60	7	9	11	13	20	2.500
	ベース	500	142	87	64	87	120	3.088
^{s}IRST	削除	230	30	34	32	52	82	2.470
	削除 (制限付)	230	30	34	32	52	82	2.470
	ベース	500	127	72	75	95	131	2.938
^{h}IAI	削除	195	20	22	31	44	78	2.292
	削除 (制限付)	195	20	22	31	44	78	2.292
	ベース	499	116	84	69	100	130	2.912
^{s}IBM	削除	84	10	15	15	18	26	2.583
	削除 (制限付)	83	9	15	15	18	26	2.554
	ベース	495	123	89	80	97	106	3.053
^{s}TALP	削除	115	12	12	23	29	39	2.383
	削除 (制限付)	112	10	11	23	29	39	2.224
	ベース	500	104	110	89	104	93	3.056
^{e}HIT	削除	211	19	41	43	57	51	2.621
	削除 (制限付)	207	17	41	42	57	50	2.604

表 6.8. 翻訳妥当性: 日英 unrestricted トラック

システム		文数	翻訳妥当性ランク別文数					翻訳妥当性
			5	4	3	2	1	ランクの平均
	ベース	500	310	85	41	27	37	4.208
[h]ATR-H	削除	53	9	12	7	9	16	2.792
	削除 (制限付)	42	1	11	6	8	16	2.357
	ベース	500	290	81	45	40	44	4.066
[s]RWTH	削除	64	12	9	10	15	18	2.719
	削除 (制限付)	57	8	8	9	15	17	2.561
	ベース	500	183	76	60	78	103	3.316
[e]UTokyo	削除	167	25	26	15	35	66	2.455
	削除 (制限付)	146	15	22	13	34	62	2.274
	ベース	500	58	70	113	133	126	2.602
[r]CLIPS	削除	268	12	18	61	81	96	2.138
	削除 (制限付)	263	11	17	60	81	94	2.125

表 6.9. 湧き出し語を含んだ訳文に対する訳語削除の実行例

(1) ボストン美術館に行くつもりです
→ I'm going to visit boston museum [after] [leaving] the [U.S.]
(2) お名前とお部屋番号を教えて下さい
→ What is the name and [departure] room number?
(3) ワインの赤はありますか → Do you have wine [free] red?
(4) あそこの出口を出てすぐです
→ Take that exit over there and [turn] right please.
(5) あちらの出口を出て下さい
→ Could you [tell] me [where] the exit over there?
(6) 目の前です → It's in front of the [station].
(7) 駅は二階にあります → Does the station is on the second floor?
(8) 駅は二階にあります → Station to your [room] is on the second floor.
(9) 食べたいですか → Do you want to eat [ice] [cream]?
(10) 野菜の料理を食べたいのですが → I'd like to have some [local] food.
(11) 野菜料理を食べたいのですが → I'd like to eat some [Chinese] food.
(12) 彼女はベジタリアンです → She is a vegetarian meal.
(13) 日本が韓国に負けています
→ There are some [similarities] [between] Japan and Korea.
(14) エアコンが煩いです → There's [no] air conditioning.
(15) エアコンがうるさいです → The air conditioner is very noisy.
(16) 明日ポーランドに行きます → I'll go to Portland tomorrow.
(17) 寝違えてしまいました → I [missed] my [station].
(18) 食中毒ではありませんか → Is this the [Washington] [library]?
(19) アトピーの症状が悪化してしまいました → I've got a [flat] [tire].
(20) あごが外れてしまいました" → The [battery] is [dead].

第 7 章

おわりに

コーパスベースの機械翻訳技術に関して，翻訳エンジン，翻訳前処理，翻訳後処理の 3 側面から新たな手法を提案し，その有効性を実証的に示した．

用例翻訳や統計翻訳といったコーパスベース翻訳では，基本的に単語や句などの小さな単位の翻訳知識を獲得しそれを使って翻訳を実行する．一方，翻訳すべき入力文とほとんど同じ文とその訳文のペアがコーパス中に存在すれば，そのペアを直接利用することで非常に良い訳が得られる．この原理に基づく文単位の用例翻訳を想定し，その実現に必要な課題に取り組んだ．用例翻訳では，入力表現に最も類似した対訳用例を抽出し，その翻訳表現の一部を変更して入力に対する訳を生成する．通常，対訳表現の単位は句である．句の組合せにより多くの入力文に対応可能であるが，各部分の誤りの合成や組合せの不整合により，不適切で不自然な訳文が生成されてしまう危険がある．一方，用例単位として文を使う場合，その危険性を抑えることができる．入力文全体に類似した文単位の用例が見つかれば，正確で自然な翻訳文を得ることができる．もちろん文は句に比べ，より長い単位であり汎用性が劣るため，文単位の用例翻訳はカバレッジの面で短所を持つ．文単位の用例を使って十分な翻訳カバレッジを得るためには，大規模な対訳コーパスを用意しなければならず，大規模コーパスから用例を検索するための効率的な検索手法が必須となる (第 2 章)．

本論文では編集距離に基づく類似度を用いた文単位の用例翻訳システムを想定し，そのシステムが大規模コーパスを扱うための効率的な検索手法を提案した．対象となる検索処理は，コーパス中の原言語文を候補文とし，入力文との距離が閾値以内で最小の候補文を全て求めることである．その目的のために全ての候補文について逐次的に距離を計算するのでは時間がかかり過ぎてしまう．そこで候補文集合の分割，単語

グラフ，A*アルゴリズムを利用した効率的な検索手法を提案した．提案手法では 2 文間の逐次的な照合は行わず，グラフ化された複数の候補文と入力文との照合を同時並行的に進める．この手法では編集距離の定義と与えられた閾値に関して検索もれはない (第 3 章).

旅行会話に関する数十万文規模の対訳コーパスを使った日英翻訳実験を通して，提案検索手法でもって実装した翻訳システムの性能を評価した．利用したコーパスは，海外旅行者向けのフレーズブックに相当する内容の日本語文とその英訳からなる．翻訳結果の品質評価のために客観評価スコアと主観評価ランクを用いた．また処理効率の評価のために，Pentium4/2GHz の通常のパーソナル・コンピュータ上での翻訳処理時間を計測した．15 万対訳からなる学習セットを使って基本的な翻訳性能を評価した結果，最高ランクに分類される訳文の割合が 71% と大きく，高い翻訳品質が示された．翻訳処理時間は，平均 0.2 秒，最大 3.3 秒であり，提案した用例検索手法により効率的な処理が実現された．コーパスサイズと翻訳性能の関係を調べるために，学習セットの大きさを 1.9 万から 30 万対訳の間で変えて性能評価を行った．結果として各指標は，コーパスサイズが大きくなるほど大きなカバレッジと高い翻訳品質が得られることを示した．一方，処理時間はコーパスサイズのほぼ 1/2 乗のオーダーに抑えられている．編集距離基準を用いた文単位の用例翻訳システムは，大規模コーパスを使うことにより高品質の翻訳能力を持ち，提案検索手法を使うことにより効率的な翻訳処理が可能となっている (第 4 章).

一般に機械翻訳システムは入力文が長くなると翻訳品質が悪くなる．第 4 章の実験でもその傾向が確認されている．そこで，コーパスベース翻訳システムを使ったより良い翻訳が可能となるように，前処理で入力文を分割することにより翻訳システムに渡す単位を適正化するアプローチをとった．文分割に関する従来の研究では，分割点周辺の N グラムなどの単語の連接の特徴に基づいた手法が多い．それに対して提案手法では，単語連接に基づく分割を補うための別指標として類似度を導入する．当分割手法では，N グラムに基づき分割点の候補を生成し，コーパスを使って計算した類似度に基づき分割点の最良の組合せを選択する．この選択は，コーパスベース翻訳システムは，学習コーパスに存在する文と類似した文は正しく翻訳することができるという仮定に基づいている．当手法の評価のため，句単位の用例翻訳，文単位の用例翻訳，統計翻訳の異なる手法による 3 つのコーパスベース翻訳システムによる英日翻訳実験を行った．実験結果は，いずれのシステムに対しても提案分割手法が有効であり，類似度の使用は翻訳品質を改善することを示している (第 5 章).

コーパスベース翻訳では，特徴的な翻訳誤りとして，入力文と関連のない語が訳文に湧き出す問題が観察される．第 4 章の実験結果からも翻訳誤りの多くがこのタイプの誤りに分類される．この問題は単語アラインメントというコーパスベース翻訳に本質的な処理に由来する．この湧き出し誤りへの対策として，後編集により自動修正するアプローチを提案した．提案手法は誤り語を自動で削除する．この手法では，まず異なる言語の単語間の対応確率を示す単語翻訳モデルを利用し，出現の期待値が閾値以下の訳語を誤り語候補として検出する．次に対訳コーパスから得られた用例を利用して誤り語候補の正しさを検証する．つまり誤り語候補が特定の類似用例に存在し，その用例において入力文と共通する語句から相対的に高い期待値が得られれば，その誤り語候補は誤りではなく正しい訳語と判断される．最終的に残った誤り語が訳文から削除される．日英および中英翻訳を対象とした複数のシステムを使った実験で，誤り語自動削除による翻訳精度向上効果が確認された (第 6 章)．

以上のように，本論文ではコーパスベース翻訳の性能向上のための手法について論じた．単語グラフを使った用例検索手法により高品質で高速な翻訳システムを実現することができた．さらに関連する翻訳誤りに対して，前処理による入力文分割，後処理による湧き出し語削除の対応策を提案し，複数の翻訳システムにおいて効果を確認した．

最後に今後の展開について簡単に触れておく．今回文単位の用例翻訳のために提案したグラフベースの検索手法は，入力文分割や湧き出し語対策においても類似度計算のために用いられている．当手法はより一般的に類似記号列を見つけるための手法としての応用が考えられる．本論文の湧き出し語対策で効果を上げた手法は，用例翻訳と統計翻訳の尺度を利用したハイブリッド処理の一種と見ることができる．今後の機械翻訳技術の一つの方向として用例翻訳，統計翻訳，ルールベース翻訳の統合が考えられる．例えば基本的な翻訳知識を人手で与え，コーパスからの学習によって知識を拡張する方法がある．コーパスベース手法とルールベース手法を組み込んだハイブリッドシステムによる様々なドメインでのより高精度な翻訳の実現の可能性がある．

謝辞

本論文をまとめるにあたり，格別なるご指導を賜りました神戸大学大学院自然科学研究科隅田英一郎教授に謹んで感謝の意を表します．また同教授には ATR 音声言語コミュニケーション研究所における機械翻訳プロジェクトのリーダとして，研究の遂行にあたって終始熱心なご討論と有益なる助言をしていただきました．重ねて感謝の意を表します．また神戸大学大学院自然科学研究科の上原邦昭教授，田村直之教授には，本論文に関して懇切なるご指導と有益なる助言をしていただきました．謹んで感謝の意を表します．また研究の進行に際し有益な議論をしていただき，さらに本研究に関するデータ収集・解析において多大なるご協力をいただきました ATR 音声言語コミュニケーション研究所の皆様に深く感謝いたします．

参考文献

[1] Y. Akiba, M. Federico, N. Kando, H. Nakaiwa, M. Paul, and J. Tsujii. Overview of the IWSLT04 evaluation campaign. *Proc. of IWSLT 2004*, pp. 1–12, 2004.

[2] Y. Akiba, T. Watanabe, and E. Sumita. Using language and translation models to select the best among outputs from multiple MT systems. *Proc. of COLING 2002*, pp. 8–14, 2002.

[3] T. Baldwin and H. Tanaka. Balancing up efficiency and accuracy in translation retrieval. *Journal of Natural Language Processing*, Vol. 8, No. 2, pp. 19–37, 2001.

[4] O. Bender, R. Zens, E. Matusov, and H. Ney. Alignment templates: the RWTH SMT system. *Proc. of IWSLT 2004*, pp. 79–84, 2004.

[5] A.L. Berger, S. A. Della Pietra, and V. J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, Vol. 22, No. 1, pp. 1–36, 1996.

[6] N. Bertoldi, R. Cattoni, M. Cettolo, and M. Federico. The ITC-irst statistical machine translation system for IWSLT-2004. *Proc. of IWSLT 2004*, pp. 51–58, 2004.

[7] J. Blatz, E. Fitzgerald, G. Foster, S. Grandrabur, C. Goutte, A. Kulesza, A. Sanchis, and N. Ueffing. Confidence estimation for machine translation. *Proc. of COLING 2004*, pp. 315–321, 2004.

[8] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, Vol. 19, No. 2, pp. 263–312, 1993.

[9] J. A. Brzozowski. Canonical regular expressions and minimal state graphs for definite events. *Proc. of Symposium of Mathematical Theory of Automata, MRI Symposia Series*, Vol. 12, pp. 529–561, 1962.

[10] M. Carl. Inducing translation templates for example-based machine translation. *Proc. of MT Summit VII*, pp. 250–258, 1999.

[11] P. Clarkson and R. Rosenfeld. Statistical language modeling using the CMU-Cambridge toolkit. *Proc. of EUROSPEECH*, pp. 2707–2710, 1997.

[12] H. T. Cormen, C. E. Leiserson, and L. R. Rivest. *Introduction to Algorithms*. The MIT Press, London, 1989.

[13] L. Cranias, H. Papageorgiou, and S. Piperidis. Example retrieval from a translation memory. *Natural Language Engineering*, Vol. 3, No. 4, pp. 255–277, 1997.

[14] G. Doddington. Automatic evaluation of machine translation quality using N-gram co-occurrence statistics. *Proc. of the HLT 2002 Conference*, 2002.

[15] T. Doi, H. Yamamoto, and E. Sumita. Example-based machine translation using efficient sentence retrieval based on edit-distance. *ACM Transactions on Asian Language Information Processing*, Vol. 4, No. 4, pp. 377–399, 2005.

[16] E. Ettelaie, K. Knight, D. Marcu, D. S. Munteanu, F. J. Och, I. Thayer, and Q. Tipu. The ISI/USC MT system. *Proc. of IWSLT 2004*, pp. 59–60, 2004.

[17] O. Furuse, S. Yamada, and K. Yamamoto. Splitting long or ill-formed input for robust spoken-language translation. *Proc. of COLING-ACL'98*, pp. 421–427, 1998.

[18] U. Germann, M. Jahr, K. Knight, D. Marcu, and K. Yamada. Fast decoding and optimal decoding for machine translation. *Proc. of ACL 2001*, pp. 228–235, 2001.

[19] A. D. Gispert and J. B. Marino. TALP: Xgram-based spoken language translation system. *Proc. of IWSLT 2004*, pp. 85–90, 2004.

[20] N. Gupta, S. Bangalore, and M. Rahim. Extracting clauses for spoken language understanding in conversational systems. *Proc. of ICSLP 2002*, pp. 361–364, 2002.

[21] S. Ikehara, S. Shirai, A. Yokoo, and H. Nakaiwa. Toward an MT system without pre-editing – effects of new methods in ALT-J/E –. *Proc. of MT*

Summit III, pp. 101–106, 1991.

[22] K. Imamura. Application of translation knowledge acquired by hierarchical phrase alignment for pattern-based MT. *Proc. of TMI-2002*, pp. 74–84, 2002.

[23] K. Imamura, H. Okuma, and E. Sumita. Practical approach to syntax-based statistical machine translation. *Proc. of MT Summit X*, pp. 267–274, 2005.

[24] G. Kikui, E. Sumita, T. Takezawa, and S. Yamamoto. Creating corpora for speech-to-speech translation. *Proc. of EUROSPEECH*, pp. 381–384, 2003.

[25] A. Lavie, D. Gates, N. Coccaro, and L. Levin. Input segmentation of spontaneous speech in JANUS: a speech-to-speech translation system. *Proc. of ECAI-96 Workshop on Dialogue Processing in Spoken Language Systems*, pp. 86–99, 1996.

[26] Y. Lee and S. Roukos. IBM spoken language translation system evaluation. *Proc. of IWSLT 2004*, pp. 39–46, 2004.

[27] D. C. Manning and H. Schütze. *Foundations of statistical natural language processing*. The MIT Press, London, England, 1999.

[28] M. Nagao. A framework of a mechanical translation between Japanese and English by analogy principle. In A. Elithorn and R. Banerji, editors, *Artificial and Human Intelligence*, pp. 173–180. North-Holland, Amsterdam, 1984.

[29] H. Ney. Stochastic modeling: From pattern classification to language translation. *Proc. of 39th ACL Workshop on DDMT*, pp. 33–37, 2001.

[30] F. J. Och and H. Ney. The alignment template approach to statistical machine translation. *Computational Linguistics*, Vol. 30, No. 4, pp. 417–449, 2004.

[31] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, Vol. 29, No. 1, pp. 19–51, 2003.

[32] K. Papineni, S. Roukos, T. Ward, and W. Zhu. Bleu: a method for automatic evaluation of machine translation. *Proc. of ACL 2002*, pp. 311–318, 2002.

[33] E. Planas and O. Furuse. Formalizing translation memories. *Proc. of MT Summit VII*, pp. 331–339, 1999.

[34] R. Rapp. A part-of-speech-based search algorithm for translation memories. *Proc. of LREC 2002*, pp. 466–472, 2002.

[35] S. Sato. CTM: An example-based translation aid system. *Proc. of COLING '92*, pp. 1259–1263, 1992.

[36] S. Sato and M. Nagao. Toward memory-based translation. *Proc. of COLING'90*, pp. 247–252, 1990.

[37] F. Sha and F. Pereira. Shallow parsing with conditional random fields. *Proc. of HLT-NAACL 2003*, pp. 213–220, 2003.

[38] M. Shimohata, E. Sumita, and Y. Matsumoto. Example-based rough translation for speech-to-speech translation. *Proc. of MT Summit IX*, pp. 354–361, 2003.

[39] H. Somers. An overview of EBMT. In M. Carl and A. Way, editors, *Recent Advances in Example-Based Machine Translation*, pp. 3–57. Kluwer Academic Publishers, Boston/Dordrecht/London, 2003.

[40] E. Sumita. An example-based machine translation system using DP-matching between word sequences. In M. Carl and A. Way, editors, *Recent Advances in Example-Based Machine Translation*, pp. 189–209. Kluwer Academic Publishers, Boston/Dordrecht/London, 2003.

[41] E. Sumita, Y. Akiba, T. Doi, A. Finch, K. Imamura, H. Okuma, M. Paul, M. Shimohata, and T. Watawabe. EBMT, SMT, hybrid and more: ATR spoken language translation system. *Proc. of IWSLT 2004*, pp. 13–20, 2004.

[42] E. Sumita and H. Iida. Experiments and prospects of example-based machine translation. *Proc. of 29th Annual Meeting of ACL*, pp. 185–192, 1991.

[43] E. Sumita, S. Yamada, K. Yamamoto, M. Paul, H. Kashioka, K. Ishikawa, and S. Shirai. Solutions to problems inherent in spoken-language translation: The ATR-MATRIX approach. *Proc. of MT Summit VII*, pp. 229–235, 1999.

[44] T. Takezawa and G. Kikui. Collecting machine-translation-aided bilingual dialogues for corpus-based speech translation. *Proc. of EUROSPEECH*, pp. 2757–2760, 2003.

[45] N. Ueffing, F.J. Och, and H. Ney. Generation of word graphs in statistical machine translation. *Proc. of Conf. on Empirical Methods for Natural Language Processing*, pp. 156–163, 2002.

[46] T. Veale and A. Way. Gaijin: A bootstrapping, template-driven approach to example-based MT. *Proc. of NeMNLP '97, New Methods in Natural*

Language Processing, 1997.

[47] S. Vogel, S. Hewavitharana, M. Kolss, and A. Waibel. The ISL statistical translation system for spoken language translation. *Proc. of IWSLT 2004*, pp. 65–72, 2004.

[48] H. Watanabe and K. Takeda. A pattern-based machine translation system extended by example-based processing. *Proc. of 36th ACL and 17th COLING*, pp. 1369–1373, 1998.

[49] T. Watanabe and E. Sumita. Example-based decoding for statistical machine translation. *Proc. of MT Summit IX*, pp. 410–417, 2003.

[50] K. Yamada and K. Knight. A syntax-based statistical translation model. *Proc. of 39th ACL*, pp. 523–530, 2001.

[51] 今村賢治，大熊英男，渡辺太郎，隅田英一郎. 統計翻訳指標を導入した構文トランスファに基づく用例翻訳. 情報処理学会研究報告 2004-NL-162, pp. 71–77, 2004.

[52] 中嶋秀治，山本博史. 音声認識過程での発話分割のための統計的言語モデル. 情報処理学会論文誌, Vol. 42, No. 11, pp. 2681–2688, 2001.

[53] 大野晋，浜西 正人. 類語新辞典. 角川書店, 1984.

[54] 菅谷史昭，竹澤寿幸，横尾昭男，山本誠一. 音声翻訳システムと人間との比較による音声翻訳能力評価手法の提案と比較実験. 電子情報通信学会論文誌, Vol. J84-D-II, No. 11, pp. 2362–2370, 2001.

[55] 隅田英一郎，佐々木裕，山本誠一. 機械翻訳システム評価法の最前線. 情報処理, Vol. 46, No. 5, pp. 552–557, 2005.

[56] 竹澤寿幸，森元逞. 発話単位の分割または接合による言語処理単位への変換手法. 自然言語処理, Vol. 6, No. 2, pp. 83–95, 1999.

[57] 山本和英. 機械翻訳における自動校正と日中翻訳への適用. 言語処理学会第 5 回年次大会, pp. 21–24, 1999.

本論文に関する原著論文

学術論文

1. 土居誉生，隅田英一郎. 単語翻訳モデルを用いた翻訳後編集による湧き出し語対策. 情報処理学会論文誌, Vol. 47, No. 6, pp. 1742–1752, 2006.
2. T. Doi, H. Yamamoto, and E. Sumita. Example-based Machine Translation Using Efficient Sentence Retrieval Based on Edit-distance. *ACM Transactions on Asian Language Information Processing*, Vol. 4, No. 4, pp. 377–399, 2005.
3. T. Doi and E. Sumita. Splitting Input for Machine Translation Using N-gram Language Model Together with Utterance Similarity. *IEICE Transactions on Information and Systems*, Vol. E88-D, No. 6, pp. 1256–1264, 2005.
4. 土居誉生，隅田英一郎，山本博史. 編集距離を使った用例翻訳の高速検索方式と翻訳性能評価. 情報処理学会論文誌, Vol. 45, No. 6, pp. 1681–1695, 2004.

国際会議

1. T. Doi, H. Yamamoto, and E. Sumita. Graph-based Retrieval for Example-based Machine Translation Using Edit-distance. *Proc. of workshop on EBMT, MT Summit X*, pp. 51–58, 2005.
2. T. Doi and E. Sumita. Splitting Input Sentence for Machine Translation Using Language Model with Sentence Similarity. *Proc. of Coling 2004*, pp. 113–119, 2004.

3. T. Doi and E. Sumita. Input Sentence Splitting and Translating. *Proc. of workshop on Building and Using Parallel Texts, HLT-NAACL 2003*, pp. 104–110, 2003.
4. T. Doi, E. Sumita, and H. Yamamoto. Adaptation Using Out-of-Domain Corpus within EBMT. *Proc. of HLT-NAACL 2003*, Companion Volume, pp. 16–18, 2003.
5. M. Paul, T. Doi, Y. Hwang, K. Imamura, H. Okuma, and E. Sumita. Nobody is Perfect: ATR's Hybrid Approach to Spoken Language Translation. *Proc. of IWSLT 2005*, pp. 55–62, 2005.
6. E. Sumita, Y. Akiba, T. Doi, A. Finch, K. Imamura, H. Okuma, M. Paul, M. Shimohata, and T. Watanabe. EBMT, SMT, Hybrid and More: ATR Spoken Language Translation System. *Proc. of IWSLT 2004*, pp. 13–20, 2004.
7. E. Sumita, Y. Akiba, T. Doi, A. Finch, K. Imamura, M. Paul, M. Shimohata, and T. Watanabe. A corpus-centered approach to spoken language translation. *Proc. of the 10th EACL*, Companion Volume, pp. 171–174, 2003.

研究会，大会

1. 土居誉生，隅田英一郎. 単語翻訳モデル駆動型の翻訳後編集. 情報処理学会研究報告 2005-NL-169, pp. 13–18, 2005.
2. 土居誉生，隅田英一郎. スラッシュ・リーディングのためのテキスト分割. 情報処理学会研究報告 2004-CE-75, pp. 25–32, 2004.
3. 土居誉生，隅田英一郎. 用例ベース翻訳 D^3 のための文分割. FIT 2002 情報科学技術フォーラム一般講演論文集, Vol. 2, pp. 181–182, 2002.