複数の単語空間を持つ LSI による言語横断検索

国分 智晴　　田中 崇　　森 辰則
横浜国立大学 工学部 電子情報工学科
E-mail: {kokubu,tanaka,mori}@forest.dnj.ynu.ac.jp

1　始めに

近年インターネットの発展などにより外国語文書を電子的に入手する機会が急激に増えており言語の壁を越えた情報検索技術である言語横断検索 (CLIR) の要求が高まってきている [1]。

言語横断検索において現在主流の方法は何らかの形で対訳辞書を用いている。そこでは辞書作りの過程で人間が対訳情報を吟味しているので、対訳辞書を用いる手法では翻訳に関する精度が良いと考えられる。その精度は対訳辞書の質や規模のみならず、その使用方法に依存するところが大きい。

一方、対訳コーパス等の言語資源から対訳に関する情報を自動的に抽出し、言語横断検索に利用する研究がある。これらの手法ではある程度の精度で検索ができることが報告されている [2]。その手法としては、対訳辞書を自動生成し、検索質問の翻訳に役立てるといった手法や、ベクトル空間法などにおける文書の中間表現の作成の際に利用する方法などがある。後者の手法として代表的なものが、 Cross Language Latent Semantic Indexing(CL-LSI) である。対訳辞書を用いずに言語横断検索が可能であるという点で魅力的であるが、 Carbonell らの報告によると、中規模コーパス (1134 対訳) を用いた場合においては、事例に基づく機械翻訳による検索質問翻訳手法が最も性能が良く、 CL-LSI 手法がこれに次ぎ、最も性能の悪かった手法が一般対訳辞書を用いた検索質問翻訳手法であった。

しかし、数万以上の対訳を含む大規模対訳コーパスを用いた場合においては、その性能は未だ不明である。そこで本稿では、 CL-LSI を大規模対訳コーパスに適用することを検討し、どの程度の精度で検索が可能であるかを検証する。

2　CL-LSI(Cross-Language Latent Semantic Indexing)

LSI は自動インデクス付けの手法である。そこではまず各文書に現れる単語の頻度を求め単語 - 文書行列を作る。各列が文書に対応するベクトルとなっており、その各成分は各単語に対応する。しかし、この方法では各単語は互いに独立した次元とみなされのそれらの間の関連は表現されない。さらに、単語 - 文書行列は一般に疎であるため、何らかのスムージングが必要とされる。 LSI ではこれを解消するために特異値分解 (SVD) を用いている。単語 - 文書行列は SVD を適用することにより、次元を縮退した単語ベクトルが得られる。これにより各単語は新しく作られた空間上に位置付けられ、単語及び文書間の類似度はそれらのベクトルの方向余弦から求められる。

上述の LSI を CLIR に利用するのが CL-LSI である [3]。 CL-LSI では対訳文書集合から単語ベクトルを作成する訓練段階と (一般に対訳ではない) 検索対象文書を LSI 空間 (単語空間) に配置し検索を行なう段階に別れる。訓練段階においては、対訳文書対を一つの文書に見たて、その文書に出現する単語の頻度を求め、単語 - 文書行列を得る。この行列に対して LSI と同様に SVD を適用し、単語ベクトルを得る (図 1)。これにより出現の仕方の似た単語が LSI 空間上において近い位置に配置されることが期待できる。特に CLIR の場合、訳語対は出現の分布が似ているので、 LSI 空間上の近い位置に配置されることが期待される。検索対象文書や検索質問は疑似文書ベクトルとして、その中に含まれる単語のベクトル和により表現される。

3　大規模コーパスにおける CL-LSI の問題点

CL-LSI 方式は文書の扱う対象領域がある程度限定されている場合に有効な手法であると考えられる。対象領域が広範囲に亙る場合に、 LSI 空間を作成するにあたって未知語の出現確率が低くなるように対訳文書対を集めるとすると、文書対の数が大きくなる。これは SVD において計算量の問題を生じさせる。 SVD は行列操作であるので、行列の次元が高くなれば、それに応じて記憶資源を消費する。よって、語彙数を高くするために非常に大きな対訳コーパスを使おうとすると、計算量の上で破綻する。

そこで我々は、訓練のために用いるコーパスを計算機の資源に併せて分割し、各々の部分コーパスから別々な LSI 空間を生成する方法を検討する。その枠組を図 2に示す。コーパスを分割すれば、 SVD が可能となり、更に、各部分コーパスの分野が限定されていれば、訳語の曖昧性の減少に役に立つと期待される。そこで検討すべき点を以下に示す。

- どのようにコーパスを部分コーパスに分割するか。
- 得られた複数の LSI 空間に対して、どのように、文書を配置するか。
- 配置された文書をどのように検索するか。

4　大規模対訳コーパス向け CL-LSI

4.1　LSI 空間の分割

文書の分野が限定されれば、文脈も自ずと限定されやすくなる。よって、個々の単語における対訳の多様

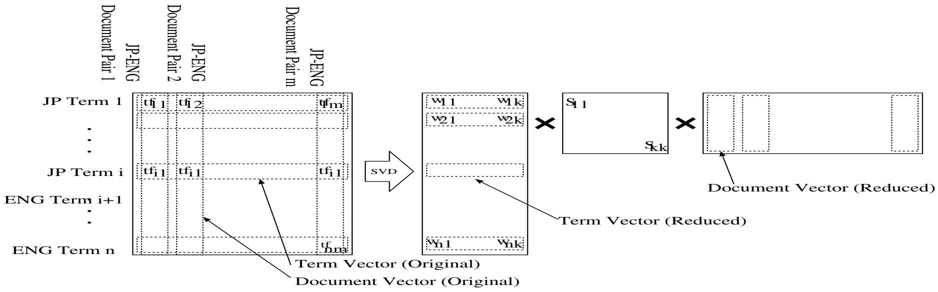

図 1: CL-LSI

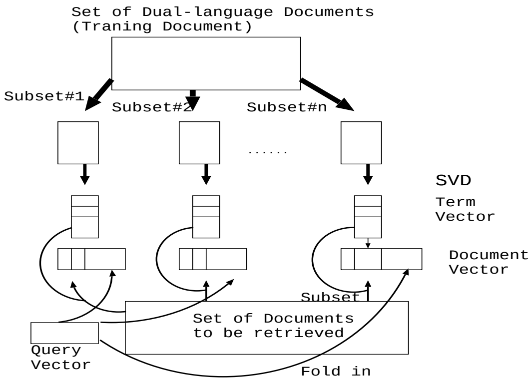

図 2: 複数の LSI 空間を持つ CL-LSI

性も軽減されると考えられる。よって、複数の LSI 空間を構築するにあたっては、類似度に従って対訳文書を複数の部分グループに分割することが有効と考えられる。自動的にこれを行なう手法としては、各種クラスタリングアルゴリズムが知られている。しかし大規模な対訳文書群に対してクラスタリングを行なうには、多大な計算機資源が必要とされる。また、いくつの文書部分集合に分割すればよいかについては、利用可能な計算機資源に依存するので最終的な調整は人間の手によるところが大きい。

そこで、別の手法を検討する。実際の文書には分野を特定するのに有効情報が付加されていることも多い。例えば学術論文を考えると、個々の文書に学会名などの分野の名前 (分野名) に関する情報が付与されているのが通常である。この情報を利用することにより、本稿では以下に述べる半自動的なクラスタリングを行なう。そこでは、同じ分野名を持つ文書対を一つのグループと考えグループを併合・分割することにより、適切な大きさのグループを作成する。

1. 対訳文書を分野名によって分類し、分野グループを作成する。
2. 各対訳文書の文書ベクトルを作成する。このベクトルは語を次元とし要素を対応する語の tfidf 値とする。
3. 同一分野グループ内の文書ベクトルの平均を求め、それを分野ベクトルとする。
4. 文書数の多い分野グループを数個、手作業で選択し、主要分野グループとする。
5. 残りの分野グループの各々について、最も類似度の高い主要分野グループに併合する。類似度は、分野ベクトルの間の方向余弦により求める。
6. 各分野グループの分野ベクトルを更新する。

4.2 検索文書の配置

本手法においては LSI 空間は分野グループによって異なるので、どの LSI 空間に文書を配置するかによって、異なる文書ベクトルが作成される。複数の LSI 空間がある状況において、文書ベクトルを作成する方法には主に、すべての LSI 空間に配置する方法と、一つの LSI 空間を選択し、そこに配置する方法が考えられる。前者の方が翻訳情報を利用するので、検索の精度が良いと考えられるが、 LSI 空間の数だけ個別の文書ベクトルが必要であり、記憶装置もその分占有してしまう。これは特に大規模文書データベースを作成する時に問題となるので、本稿では選択した一つの LSI 空間のみに配置する。また、この方式では、文書ベクトルを配置する LSI 空間を適切に選択する方法を考えなければならない。ある文書を配置先は、訳語選択の分野依存性から、同一の分野の対訳から作成された LSI 空間であることが望ましい。よって、各文書を最も類似する分野ベクトルを持つ分野の LSI 空間に配置する。

文書を LSI 空間に畳み込む方法としては次の式を用いる。

$$\mathbf{D} = \sum_{\mathbf{T_i} \in \mathbf{D}} \mathbf{tf}(\mathbf{T_i}, \mathbf{D})\mathbf{idf}(\mathbf{T_i})\mathbf{T_i}$$

D	:	文書 D のベクトル
$tf(T_i, D)$	:	文書 D 中の語 T_i の頻度
$idf(T_i)$	:	語 T_i の idf 値 $\log \frac{N}{df(T_i)} + 1$。$df(T_i)$ は T_i の文書頻度
$\mathbf{T_i}$	:	語 T_i の ベクトル

4.3 複数 LSI 空間での文書検索

CL-LSI 方式では検索質問も他の文書と同様に LSI 空間上のベクトルとして表現され、各文書は、検索質

問ベクトルと間の類似度に基づき順位づけされるが、我々の方法では LSI 空間が複数あるので、以下の手順により文書検索を行なう。

1. 検索質問をすべての文書と比較するために、各 LSI 空間に対して、検索質問ベクトルを一つ作成する。
2. 各 LSI 空間毎に、検索質問ベクトルとすべての文書ベクトルの間の類似度を計算する。
3. その類似度を、複数の LSI 空間に亙って、降順に整列することにより文書に対する順位づけを行なう。

5 対訳コーパスの分割による未知語の問題

CL-LSI 方式では対訳コーパス中に現れない単語については、その対訳情報が得られないので、文書ベクトルを作成するときに無視される。よって、検索対象文書中には現れるが、対訳コーパスに現れない語が検索質問中に含まれるときには、検索精度が低下する。これは不可避であるが本方式には、もう一つの未知語の問題がある。これは、対訳コーパスを分割することにより生じたものである。

コーパスを分野毎に分割した場合、関連する部分コーパスのみに現れ、他の部分コーパスに現れないような語が存在しうる。すなわち、 LSI 空間毎に未知語が異なっている。そのため、文書検索において期待どおりの結果が得られないことがある。

例として、検索質問から得られた T_a、 T_b、 T_c という 3 つの語で LSI 空間 TS_1 と TS_2 中の文書を検索する場合を考える。 LSI 空間 TS_1 には、語 T_a が存在し、 T_b、 T_c が未知語となっているとする。そこには、 T_a のみが含まれる文書 D_1 が配置されているとする。また、 LSI 空間 TS_2 には、語 T_a、 T_b、 T_c のすべてが存在するとし、ここに、 T_a、 T_b が含まれる文書 D_2 を配置するとする。

ここでは当然、 D_1 よりも D_2 を検索したく、 D_2 の類似度の方が大きくなることを期待する。しかし、上述の状況においては、全く逆で、 D_1 の類似度のほうが D_2 よりも大きくなってしまう。これは TS_1 での類似度計算は、 T_a のみについて行なわれるので、検索質問があたかも T_a であると見なされるためである。我々の望む類似度計算を行なうためには、検索質問中の未知語を単に無視するのではなく、類似度を低下させる要因として適切に扱わなければならない。我々はその一つとして、未知語に対応する新しい次元を一つ LSI 空間に導入する方法を提案する。この方法は、次に述べる手順で LSI 空間を拡張し、未知語を既存のどの単語ベクトルとも直行するベクトルとして扱うことにより、類似度に対する補正を行なうものである。ある LSI 空間が n 次元空間に表現されているとする。すると、単語は $(w_1, ..., w_n)$ なるベクトルになる。この空間に対して、新たな次元を導入し、空間を (n+1) 次元とする。このとき、既存の単語については，$(w_1, ..., w_n, 0)$ とし、一方、検索質問に現れるすべての未知語を、 $(0, ..., 0, 1)$ とする。このように新たに構築された空間での類似度は次のように検索質問に文書に含まれている未知語の分だけ低く見積もられる。補正後の文書ベクトル D' と検索質問ベクトル Q' の間の類似度 $sim(D', Q')$ は、

$$sim(\mathbf{D}', \mathbf{Q}') = \frac{\mathbf{D}' \cdot \mathbf{Q}'}{|\mathbf{D}'||\mathbf{Q}'|}$$

であるが、ここで、以下の式が成り立つ。

$$\begin{aligned} |\mathbf{Q}'| &= |\mathbf{Q} + \mathbf{Q_u}| \\ &= \sqrt{|\mathbf{Q}|^2 + (\sum_{\mathbf{T_i} \in \mathbf{Q_u}} \mathbf{tf}(\mathbf{T_i}, \mathbf{Q_u})\mathbf{idf}(\mathbf{T_i}))^2} \end{aligned}$$

$\mathbf{D}$、 $\mathbf{D}'$: 補正前後の文書ベクトル
$\mathbf{Q}$、 $\mathbf{Q}'$: 補正前後の検索質問ベクトル
$\mathbf{Q_u}$: 検索質問中の未知語部分 Q_u のベクトル

よって、以下の式となる。

$$sim(\mathbf{D}', \mathbf{Q}') = \frac{\mathbf{D} \cdot \mathbf{Q}}{|\mathbf{D}|\sqrt{|\mathbf{Q}|^2 + (\sum_{\mathbf{T_i} \in \mathbf{Q_u}} \mathbf{tf}(\mathbf{T_i}, \mathbf{Q_u})\mathbf{idf}(\mathbf{T_i}))^2}}$$

6 評価実験

6.1 LSI 空間の分割

LSI 空間を分野毎に作成して検索を行なうことにより検索精度がどの程度向上するかを、メイト検索[1]実験により評価した。まず NTCIR1 の言語横断タスクから得られた学会情報付の対訳技術文書 (要約) 6000 対を訓練コーパスとして用いた。その訓練コーパスを学会情報を基に 3 つに分割し、部分 LSI 空間を作成し提案手法により検索した場合を、すべての訓練コーパスにより一つの LSI 空間を作成、検索を行なった場合と比較した。検索対象として訓練コーパスとは別の対訳文書 (3000 対) を用いた。その結果を表 1に示す。これによると、 LSI 空間を分割することにより同等もしくは若干の精度の向上が見られた。

表 1: 分割 LSI 空間の精度評価

LSI 空間	1 位 (%)	3 位 (%)
全体	58.2	75.7
分割	47.8	63.9
分割 補正後	59.4	78.2

6.2 NTICR2 における実験

NTICR2 において言語横断タスクが行なわれた。このタスクは多様な検索手法の相互比較し、どの手法がどのような効果をもたらすかなどについて、検証す

[1]検索対象文書集合から文書を一つ選択し、その対訳を検索質問とする。このとき、元の文書が何位で検索されるかをみることにより、検索精度を検証する手法。

るために開催されている。検索規模は非常に大きく、提案手法が大規模な検索タスクでどの程度の精度となるのかを評価できる。訓練コーパスとしてはNTCIR1で使用された日英技術文書要約約38万件を使用することが可能で、検索対象は日英技術文書要約約70万件であった。

本実験では上記コーパスから日英の対訳対が得られた約18万文書対を用いた。検索トピックとしては日英各々49件あり、我々の実験は、DESCRIPTIONフィールド(1文程度)単体を検索質問とした場合と、DESCRIPTIONとNARRATIVE(要約文書程度)を合わせたものを用いた場合で行なった。索引づけには単単語ならびに複合語を用いた。複合語の認定には、すべての言語で一貫して扱える手法として、C value に基づく方法を用いてる[4]。日本語文書については、形態素解析器JUMAN 3.61により、単語切り出しならびに品詞付与を行なった。複合語の抽出は次の二段階で行なった。まず、対訳コーパスより、語を単位とするサフィックスアレイを作成し、単語列の出現頻度を求めた。そしてある閾値TH_f以上の出現回数をもつ単語列を複合語の候補とした。次に各複合語候補に対して、C value を計算し、その値がある閾値TH_c以上のものを複合語として認定した。今回の実験では、プログラムの制約上、NTCIR1の対訳コーパスを11に分割し、各部分集合において、$TH_f = 5, TH_c = 5$の条件件の下で、上記手続きを行なった。なお、複合語の構成要素も索引づけに用いている。結果を表2に示す。

表 2: 全ての検索質問に対する平均適合率、R適合率

	Average precision	R-Precision
J-E-Desc	0.0533	0.0635
J-E-Desc 補正後	0.0666	0.0786
補正による改善	24.9 pt	23.8 pt
J-E-Desc-Nar	0.0868	0.1031
J-E-Desc-Nar	0.0940	0.1096
補正による改善	8.3 pt	6.3 pt
E-J-Desc	0.0512	0.0705
E-J-Desc	0.0610	0.0839
補正による改善	19.1 pt	19.2 pt
E-J-Desc-Nar	0.0609	0.0876
E-J-Desc-Nar	0.0736	0.1018
補正による改善	20.8 pt	16.2 pt

本方式ではどの訓練コーパス上にも出現しない語については、単語ベクトルが存在しないのでその語が検索質問に出現した場合に検索精度が悪くなる。そこで未知語のない検索質問においてどの程度の精度が見込まれるかを別途評価した。結果を表3に示す。

絶対的な性能評価の観点からすると、やはり人手で構築した対訳辞書に基づく手法に比べ、検索精度がかなり低いといわざるをえない。NTCIR2における最もよいシステムの平均適合率が 30%を越えているのに対し、我々の手法では約10%である。しかし、ある程度の規模の対訳コーパスがあれば、大規模な言語横断検索も可能であるということが確認された。また、我々の導入した補正手法の効果も確認される。特にそれはDESCRIPTIONフィールドだけを用いた場合のほうが、DESCRIPTIONならびにNARRATIVEフィールドを用いた場合よりも顕著である。これは、短い検索要求のほうが未知語の影響が現れやすいためである。

表 3: 未知語のない検索質問に対する平均適合率、R適合率

	検索質問数	Average precision	R-precision
J-E-Desc	43	0.0600	0.0704
J-E-Desc 補正後	43	0.0743	0.0870
補正による改善		23.8 pt	23.6 pt
J-E-Desc-Nar	31	0.1032	0.1206
J-E-Desc-Nar 補正後	31	0.1094	0.1307
補正による改善		6.0 pt	8.4 pt
E-J-Desc	43	0.0579	0.0782
E-J-Desc	43	0.0692	0.0942
補正による改善		19.5 pt	20.4
E-J-Desc-Nar	39	0.0738	0.1025
E-J-Desc-Nar	39	0.0872	0.1187
補正による改善		18.1pt	15.8 pt

7 おわりに

本稿では、既存の対訳コーパスのみを翻訳の情報として用いる情報検索手法として、CL-LSI手法に注目した。我々は，これを大規模対訳コーパスに適用する手法として、複数のLSI空間を併用する方法を提案した。また、LSI空間毎の未知語に起因する検索精度低下について検討し、対処方法を提案、その効果をNTCIR2のタスクにおいて確認した。

一方、今回の実験ではLSI空間の分割による精度向上がメイト検索ではない実際の検索の場面において、精度向上に役立っているかは不明なままなので別途実験を行ない確認する必要がある。またGVSMなどの他の類似方法で大規模コーパスを用いた場合との比較をしていきたい。

参考文献

[1] 菊井玄一郎. 言語の壁を越えて文書を検索する. 人工知能学会誌, Vol. 15, No. 4, July 2000.

[2] J. G. Carbonell, Yang. y., R. E. Frederking, R. D. Brown, Y. Geng, and D. Lee. Translingual information retreival. *A Comparative Evaluation, in Proceedings of IJCAI-98*, 1997.

[3] Susan T.Dumais, Todd A.Letsche, Thomas K. Landauer, and Michael L.Littman. Automatic cross-linguistic information retrieval using latent semantic indexing. *AAAI Spring Symposium on Cross-Language Text and Speech Retrieval,*, 1997.

[4] Katerina T. Frantzi and Sophia Ananiadou. Extracting nested collocatins. *COLING-96*, 1996.