修士論文

特許検索における質問意図の曖昧化

48-156229　**胡瀚林**

指導教員　中川裕志 教授

2017 **年** 1 **月**

東京大学大学院情報理工学系研究科数理情報学専攻

概要

企業が特許を取る前に，類似な特許が既に存在するかを確かめるために特許データベースを検索する必要がある．しかし，検索の質問から企業秘密が漏洩する可能性がある．ウェブテキスト検索の質問からユーザーの検索意図を守る手法が多数存在している．その中真の質問と同時にダミー質問を提出する質問曖昧化手法が一番効率的、現実的である．本論文では特許検索における既存な質問曖昧化手法 [1, 2, 3] を実装し，類似度攻撃 [4] で特許データベースにおける既存手法の安全性を評価した．

また、類似度攻撃 [4] を含め、多くの既存な質問曖昧化に対する攻撃手法は攻撃者が質問者の事前情報を持つと仮定する．本論文では事前情報なしの攻撃手法を提案し，その攻撃手法に対応できる既存な質問曖昧化の改良と新たな質問曖昧化手法を提案する．

目次

第 1 章

はじめに

テキスト検索をするとき，検索質問をサーバー側に渡さなければならない．しかし，検索質問から質問者の情報が漏洩する危険があることが AOL 事件 [?] より証明された．特許検索の場合は検索質問が研究開発動向など企業秘密を含んでいるため，一般的なウェブ検索の質問者より質問のプライバシー問題を重視している．そのような問題を解く様々な手法が存在している．[] や [] などの IP アドレスの匿名化メカニズムは登録情報が必要な検索サーバーに対応できない．また検索質問のみから質問者を一意に特定されてしまう可能性がある [?]．プライベート情報検索 (Private Information Retrieval)[] は計算量的安全性を持つが，サーバー側で大量の計算が必要であるため実用するのは難しい．曖昧化検索 (Obfuscation Search)[] は真の質問を分析し適切な $K - 1$ 個のダミー質問を生成し真の質問と同時に検索する．安全性が弱いが，効率よく質問者の検索意図を守ることができる．

本論文の構成は次の通りである．第二章では特許文章と特許検索の特徴を述べる．第三章では既存な質問曖昧化メカニズム [1, 2, 3] を述べる．第四章では曖昧化メカニズムがよく用いる意味分析手法を述べる．第五章では既存な攻撃手法 [4] を述べ，[4] の改良と新たな攻撃手法を提案する．第六，七章では新たな質問曖昧化手法を提案する．最後に, 第八章で評価実験を述べ，第九章で全体をまとめる．

第 2 章

特許

特許検索質問のプライバシーを保護する手法を説明する前に特許検索と特許そのものを簡単に紹介する必要がある．特許法第 1 条には，「この法律は、発明の保護及び利用を図ることにより、発明を奨励し、もつて産業の発達に寄与することを目的とする」とある．特許制度は，発明者には一定期間，一定の条件のもとに特許権という独占的な権利を与えて発明の保護を図る一方，その発明を公開して利用を図ることにより新しい技術を人類共通の財産としていくことを定めて，これにより技術の進歩を促進し，産業の発達に寄与しようというものである．[?] 特許を取るには以下の条件を満たさなければならない: 新規性:公知の発明と同様の発明は特許を受けることができない; 進歩性:先行技術に基づいて容易に発明をすることができる発明は特許を受けることができない．単一性:発明の単一性の要件を満たさない二以上の発明は一つの願書で出願することができない.

特許を受けようとする発明を特定するために特許請求の範囲を記載する必要がある.

図 2.1 で表した例のように，特許の請求項は特定の書き方がある．誤解を招かないように技術用語は、学術用語を用いる．また，一般的な文章は単語をなるべく重複しないようにする一方，特許文章は単語を全体を通じて統一して使用する.

2.1 特許分類

特許の一つ特徴は全ての特許が人の手によって分類されている．特許分類を用いることより検索する特許文章が減り，似たようなキーワードを含むが分類が違う特許文章を排除することができる．今最も使われている特許分類が世界知的所有権機関 (WIPO) による管理されている国際特許分類 (IPC) である．国際特許分類は階層構造であり，一番上の階層は A から H までの 8 個のセクションである．セクション以下は**??**に表したように四つの階層に分類されている.

2.2 特許検索

(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2016-208844
(P2016-208844A)
(43)公開日　平成28年12月15日(2016.12.15)

(51)Int.Cl.	F I		テーマコード（参考）
A01B 35/04 (2006.01)	A01B 35/04	E	2B034
	A01B 35/04	B	

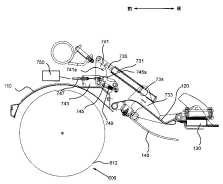
審査請求 未請求　請求項の数 4　OL　（全 13 頁）

(21)出願番号　特願2015-92186(P2015-92186)
(22)出願日　平成27年4月28日(2015.4.28)

(71)出願人　390010836
小橋工業株式会社
岡山県岡山市南区中畦６８４番地
(74)代理人　110000408
特許業務法人高橋・林アンドパートナーズ
(72)発明者　河原　文雄
岡山県岡山市南区中畦６８４番地　小橋工業株式会社内
Fターム(参考)　2B034 AA03　BA06　BB01　BB02　EA02　EB06　EB33　JA06

(54)【発明の名称】農作業機

(57)【要約】
【課題】代かき作業機を昇降させる必要がない場面において、オート装置が代かき作業機を昇降させることを防止する。
【解決手段】本発明の一実施形態に係る農作業機は、耕耘作業を行うロータリ作業部を回転自在に支持する機体と、機体に設けられ、ロータリ作業部の上部を覆うカバー部と、カバー部の後端部に回動可能に支持されたエプロンと、エプロンの背面に取り付けられた支持部材と、カバー部に取り付けられ、エプロンのロック状態とフリー状態を切り替え可能なエプロン回動制御部と、を備え、エプロン回動制御部は、後端部が支持部材に対し取り付けられたロッド部と、ロッド部の前端部を回動自在に支持し、被係合部を有する第１アーム部と、カバー部に回動自在に支持され、係合部を有する第２アーム部と、第２アーム部を回動させる駆動部とを有し、係合部は、第１アーム部が回動するときに、被係合部の回動を規制するように構成されてもよい。
【選択図】図１



【特許請求の範囲】
【請求項１】
耕耘作業を行うロータリ作業部を回転自在に支持する機体と、
前記機体に設けられ、前記ロータリ作業部の上部を覆うカバー部と、
前記カバー部の後端部に回動可能に支持されたエプロンと、
前記エプロンの背面に取り付けられた支持部材と、
前記カバー部に取り付けられ、前記エプロンが自由に回動できない状態であるロック状態と前記エプロンが自由に回動できる状態であるフリー状態とに切り替えることができるエプロン回動制御部と、を備え、
前記エプロン回動制御部は、後端部が前記支持部材に対して、摺動可能に取り付けられたロッド部と、
前記ロッド部の前端部を回動自在に支持し、前記カバー部に対して回動自在に支持されて、被係合部を有する第１アーム部と、
前記カバー部に回動自在に支持され、係合部を有する第２アーム部と、
前記第２アーム部を回動させる駆動部とを有し、
前記係合部は、前記ロッド部が前方に移動するのに伴い前記第１アーム部が回動するときに、前記被係合部の回動を規制することを特徴とする農作業機。
【請求項２】
前記被係合部は、ピン部材であることを特徴とする請求項１に記載の農作業機。
【請求項３】
前記駆動部は、ワイヤとワイヤ制御部を含むことを特徴とする請求項１又は請求項２に記載の農作業機。
【請求項４】
耕耘作業を行うロータリ作業部を回転自在に支持する機体と、
前記機体に設けられ、前記ロータリ作業部の上部を覆うカバー部と、
前記カバー部の後端部に回動可能に支持されたエプロンと、
前記エプロンの背面に取り付けられた支持部材と、
前記カバー部に取り付けられ、前記エプロンが自由に回動できない状態であるロック状態と前記エプロンが自由に回動できる状態であるフリー状態とに切り替えることができるエプロン回動制御部と、を備え、
前記エプロン回動制御部は、後端部が前記支持部材に対して、摺動可能に取り付けられたロッド部と、
前記ロッド部の前端部を回動自在に支持し、前記カバー部に対して回動自在に支持されて、係合部を有する第１アーム部と、
前記カバー部に回動自在に支持され、被係合部を有する第２アーム部と、
前記第２アーム部を回動させる駆動部とを有し、
前記被係合部は、前記ロッド部が前方に移動するのに伴い前記第１アーム部が回動するときに、前記係合部の回動を規制することを特徴とする農作業機。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、農作業機に関する。特に、本発明は、エプロンが自由に回動できない状態であるロック状態とエプロンが自由に回動できる状態であるフリー状態とに切り替えることができるエプロン回動制御部を備える農作業機に関する。
【背景技術】
【０００２】
耕耘ロータにより耕耘された耕土を整地するエプロン（第１整地板）とエプロンの後部に上下方向に回動自在に設けられて耕土表面を均平にするレベラ（第２整地板）を備える農作業機、例えば、代かき作業機は、一般に、走行可能な走行機体の後部に三点リンク連結機構を介して昇降可能に連結されて、走行機体の前進走行とともに進行しながら代かき

図 2.1. 特許文章例

セクション:A	健康および娯楽
サブセクション：61	医学または獣医学:衛生学
クラス: C	歯科:口腔または歯科衛生
メイングループ:5	歯の充填または被覆
サブグループ:08	歯冠:その製造; 口中での歯冠固定

表 2.1. 国際特許分類例:A61C 5/08

検索タイプー	検索対象 (specification)	検索目的
技術水準調査 (State of the Art Search)	イデア	自分の発明に関連する背景知識を得る
新規性調査 (Novelty Search)	特許文章	特許登録の可能性を判断する
侵害調査 (Infringement Search)	商品と 商品に関連する技術	権利侵害とならないかを判断する

表 2.2. 特許検索タイプー

符号	意味
N	辞書中の単語の数
$W = \{1, 2, 3, \ldots, N\}$	単語集合
M	コーパス中の文書の数
$D = \{1, 2, 3, \ldots, M\}$	文章集合
K	トピック数
$T = \{1, 2, 3, \ldots, K\}$	トピック集合
$\ell_i = \{t_1, t_2, \ldots, K\}$	単語 i のトピックベクトル
ℓ	質問のトピックベクトル

表 2.3. 表記法

第 3 章

曖昧化検索

曖昧化検索は質問者が検索したい真の質問と質問者側で生成したダミー質問を一緒に検索サーバーに提出し，真の質問がどれかを曖昧化するものである．本論文では以下のモデル [] を用いて既存な曖昧化検索メカニズムを分析する．質問者 Alice がとある検索サーパーに質問を出して手に入れたい情報を検索し，検索サーバーが semi-honest な攻撃者であることを仮定する．

質問が単語の集合であり，質問の定義域を単語集合の冪集合にする．

定義 1. ユニバーサル質問集合 Q．W を全ての単語の集合とする．ユニバーサル質問集合 Q とは W の冪集合である，つまり

$$Q = P(W) = \{A | A \subset W\} \tag{3.1}$$

Alice のプロフィールを多項分布と仮定し，Alice が持つ真のプロフィールを X とする．

定義 2. 質問者のプロフィール X．T を全てのトピックの集合とする．質問者のプロフィール X とは

$$X = \{x_i | i \in T\} \tag{3.2}$$

x_i は質問者がトピック i に対して持つ興味の強さを表す．

曖昧化検索メカニズムは Alice のコンピュータで実行する．曖昧化検索メカニズムが意味分析ツール SA を用いて真の質問 q_R を分析しダミー質問 q_D を生成する．生成したしダミー質問 q_D と真の質問 q_R を 1 つの質問グループにし，検索サーバーに提出する．質問 q とトピック t の関係を表す関数は以下のように定義する，

定義 3. 質問-トピックスコア関数:$rscore_{SA}$．T を全てのトピックの集合とする．質問 q とトピック t の関係を表す関数とは

$$rscore_{SA} : Q \times T \to \mathbb{R} \tag{3.3}$$

定義 4. 質問 q のメイントピック:$\delta_{SA}(q)$．T を全てのトピックの集合とする．質問 q のメイ

ントピック δ_q とは

$$\delta_{SA}(q) = \underset{t \in T}{\operatorname{argmax}}\, rscore_{SA}(q,t) \tag{3.4}$$

次は質問 q のトピックベクトルを定義する．質問 q のトピックベクトル $vec_{SA}(q) = (rscore_{SA}(q,t_1),\ldots,rscore_{SA}(q,t_{|T|})$ とは q と全てのトピック t_i の質問-トピックスコア関数 $rscore(q,t_i)$ を要素として持つ $|T|$ 次元ベクトルである．質問のトピックベクトルを使って質問間の関係を評価することができる．

検索サーバーが Alice からもらった質問をすべて記録し，その質問たちを分析し得るプロフィールを Y にする．

定義 5. 質問比較関数:C．質問比較関数 $C : Q \times Q \to \mathbb{R}$ を以下のように定義する

$$C_{SA}(q_1,q_2) = \frac{(vec_{SA}(q_1)\cdot vec_{SA}(q_1))}{||q_1||||q_2||} \tag{3.5}$$

曖昧化検索は 3 つ違うレベルな目標がある．まずは質問そのものの曖昧化である．質問者が検索した真の質問 q_R はどの質問であるかをわからないようにする．2 つ目は質問意図の曖昧化である．質問者が検索したいものは何であるかをわからないようにする．最後は質問者のプロフィール X の曖昧化である．Y から質問者が興味を持つトピックは何であるかをわからないようにする．

質問の曖昧化ができたとしても質問意図の曖昧化ができると限れない．林檎とリンゴの 2 つ質問から真の質問を確定することができないが，質問者が林檎について検索したいことが確定できる．同じように林檎と梨の 2 つ質問から質問者が検索したいを確定することができないが，質問者が果物に興味を持つことが確定できる．本論文では質問意図の曖昧化をメインにする．

次に検索質問のプライバシー保護の代表的な手法，否認可能検索 (PDS)[1]，質問者のプライバシーを保護する質問加工法 (ETSQ)[2]，質問意図を曖昧化するキーワード検索 (OMTI)[3] を紹介する．

3.1　否認可能検索

否認可能検索という概念を提出したのは [] である．つまり， サーバーは特定なユーザーが特定の時間に提出した一連の質問 $L = q_1, q_2, \ldots, q_K$ のログを持つと仮定する．ログにアクセスしたある人が真の検索質問が q_i だと証明したいとき， L の中の任意の質問 q_j が真の質問となる確率が同じ $1/K$ だと証明できる．以下に否認可能検索を定義する．

定義 6. k − 否認可能検索質問 q をユーザーが入力した質問とする．ダミー質問生成システム D が k 個の質問を含んでいる質問集合 $D(q_u) = \{q_1,\ldots,q_k\}$ を出力しサーバーに提出する．$D(q_u)$ が以下の性質を持つなら，$D(q_u)$ を PD-質問集合といい，D を k −否認可能検索という

1. $\exists q_i \in D(q_u), q_i$ と q_u が意味的に近い

2. $\forall q_j \in D(q_u), D(q_j) = D(q_u)$
3. $\forall q_j \in D(q_u), q_j$ が違うトピックに含まれる
4. $\forall q_j \in D(q_u), q_j$ が同じような尤もらしさを持つ

[1] では事前に文書集合から高頻度な単語と単語ペアをシード質問として抽出し，潜在意味分析 (LSA)[] を用いてシード質問をトピック空間にマップし，トピック空間に距離が近いシード質問をクラスタリングして標準質問と PD-質問集合を構築する．検索する場合は，ユーザーが検索したい質問の代わり, 事前に用意した標準質問集合からトピック空間において質問者が検索したい真の質問と最も近い標準質問が属する PD-質問集合をサーバーに提出し，サーバーから検索結果を得，質問者側で真の質問を用いて検索結果をフィルタリングする．以下でこの流れを具体的に述べる．

3.1.1 シード単語と標準質問

システムが生成した質問は通常は使わない単語の組み合わせを使うことがある．攻撃者がこのような質問をダミー質問と判定し、真の質問を特定する可能性があるため，PDS は標準質問と PD-質問集合を事前に構築する．そのため，Q の中の全ての質問をカバーすることは不可能である．PDS の目標は妥当な再現率を得ることであるため，高頻度な単語だけを使うことは適当だと考えられる．

Algorithm 1 標準質問の構築

Input: シード質問集合 S

1: $Q_C / Leftarrow \phi$
2: Kdtree を構築し S の全ての要素を追加する
3: **for all** $s_i \in s$ **do**
4: Kdtree を用いて s_i と最も近いシード質問 c_1, c_2 を選ぶ
5: $cquery = s_i \cup c_1 \cup c_2$
6: **if** $cquery \notin Q_c$ **then**
7: $Q_C = Q_C \cup \{cquery\}$

Output: 標準質問の集合 Q_C

まず単語・文書行列に頻出パターンマイニング [] を用いて Δ 回以上に表れた単語と連続する単語からなる単語ペアをシード質問として抽出し，トピック空間にマップする．シード質問はユーザーの意図を適切に表さないことが多いため，PDS では意味的に近いシード質問をグループにして標準質問にする．アルゴリズム 1 ではこの流れを具体的に説明する．このステップの計算量は $O(NlogN)$ となる．ここで N はシード質問の数である．

3.1.2　PD-質問集合の構築

PD-質問集合を構築するには，トピックは異なるが尤もらしさが近い標準質問を同じ質問集合に集めれば良い．そのため，多様性と尤もらしさを計算する方法を提案する必要がある．多様度ではトピック空間の中の距離で評価する．人間が作った質問と比較するため，合理的な大きさを持つ質問ログ $Q_L = \{q : q \in Q\}$ にアクセスできると仮定する．Q_C と同様に Q_L もトピック空間にマップし，標準質問の近傍の中の Q_L の要素数で標準質問の尤もらしさを計算する．近傍に多くの Q_L に含まれる質問がある標準質問を尤もらしさが高いとする．

次は 3 つの部分の和となる標準質問間の関係を評価する関数を定義する．質問 q_1 と質問 q_2 のユークリッド距離 $edist(q_1, q_2)a$ とは，

$$edist(q_1, q_2) = \sqrt{\sum_{i \in T}(vec_{LSA}(q_1)[i] - vec_{LSA}(q_2)[i])^2} \tag{3.6}$$

である．ユークリッド距離が遠い質問が異なるトピックに含まれると考えられる．質問 q の強度とは，

$$||q|| = \sqrt{\sum_{i \in T}(vec_{LSA}(q)[i])^2} \tag{3.7}$$

である．質問 q の近傍中の質問数 $nhc(q)$ とは，

$$nhc(q) = count(vec_{LSA}(q), Q_L, HCUBE(vec_{LSA}(q), \vec{(}\delta))) \tag{3.8}$$

である．ここで Q_L は質問ログ，$HCUBE(vec_{LSA}(q), \vec{(}\delta))$ は $vec_{LSA}(q)[i] \pm \delta[i]$ となる超立方体である．$nhc(q)$ は超立方体中で Q_L に属するベクトルの数を返す．

定義 7. 質問間の評価関数:dis.

$$dis(q_1, q_2) = (1 - \frac{edist(q_1, q_2)}{\alpha}) + \frac{|||q_1|| - ||q_2|||}{\beta} + \frac{|nhc(q_1) - nhc(q_2)|}{\gamma} \tag{3.9}$$

ここで，α は Q_C に属する全ての質問ペア間の最大のユークリッド距離，β は質問ペア間の最大の強度差で，$gamma$ は質問ペア間の最大の近傍中の質問数の差である．

したがって，近傍中の質問数と強度の差が小さく，トピック空間中の距離が遠い質問ペアの評価関数の値が低くなり，一つの PD-質問集合に入れるべきである．

次では，質問集合間の評価関数を定義する．$A = a_1, \ldots, a_n$ と $B = b_1, \ldots, b_m$ を 2 つ質問集合とする．A，B 間の評価関数とは，

$$dis(A, B) = (1 - \alpha_1/\alpha) + \beta_1/\beta + \gamma_1\gamma \tag{3.10}$$

である．ここで，$\alpha_1 = \min_{i,j}(edist(a_i, b_j))$ は 2 つの質問集合に属する質問ペア間のユークリッド距離の最小値であり，$\beta_1 = |\frac{\sum_i ||a_i||}{n} - \frac{\sum_j ||b_j||}{m}|$ と $\gamma_1 = |\frac{\sum_i nhc(a_i)}{n} - \frac{\sum_j nhc(b_j)}{m}|$ は質問集合の強度と近傍中の質問数の平均数の差である．

3.1.3 凝集型クラスタリング

PDS では，まず質問ペアを要素とするレベル 1 集合 L_1 を生成する．Q_C に属する全の質問ペア間の評価関数の値の行列を計算し，評価関数の値が小さいから大きい順で質問ペアを L_1 に加える．質問ペア (q_i, q_j) に対し,q_i か q_j は評価関数の値がもっと小さいペアに属する可能性がある．その場合，q_i か q_j がすでに L_1 にあることとなり, 次に評価関数の値が小さいな質問ペアを選ぶ．選んだ質問ペアをマージし，次のレベルの集合 (L_2,L_3，etc) を作る．マージステップはレベル変数 l が $log_2 k$ になるまで続ける．したがって，最終レベルの集合中の質問クラスターの大きさが k となり, オーバーラップがないと保証する．

3.1.4 PD-質問集合の使用

ユーザー質問 q_u に近い標準質問を探すため，q_u を意味区間にマップし，$C(q_u, q_c)$ が一番大きい標準質問 q_c を選び，q_c が属する PD-質問集合をサーバーに提出し，クライエント側でユーザー質問を用いて検索結果をフィルターする．一定な再現率を得るため，普段の検索より多くの文書を手に入れる必要があるが，フィルターステップがこの影響をなくす．

ユーザー質問の全ての単語が PD-質問集合を構築するために使った単語リストに含んでいないなら，ユーザー質問を意味区間にマップすることは不可能である．しかし，単語量が十分大きいなら，そのような状況を発生する可能性は低いと考えられる．また，(十分大きな) 単語リストに含んでいない単語はユーザーの意図を漏洩するリスクが高い．ユーザーがそのような質問を提出したとき，ユーザーに危険性を警告し，検索しないようにすることが考えられる．

3.1.5 プライバシー分析

PDS では真の質問 q_u の代わりに真の質問と一番類似した質問 q_c を含んでいる質問セットをサーバーに送る。q_c が真の質問の大半な結果を検索できると考えられ，質問セット中の他の質問をダミー質問に見なす．そのため, このメカニズムは検索の精度と再現率に影響を大きく与える．また，質問の長さの増加と伴って質問の可能な組み合わせが指数的に増加するため，実践的には特許検索など長い質問が多いテキスト検索と質問拡張に対応できない．

3.2 質問者のプライバシーを保護する質問加工法

今テキスト検索エンジンの大半が類似検索である．全ての質問単語を含んでいる文章しか検索できないキーワード検索と違い，類似検索は文章と質問の関連性を計算し文章にスコアをつける []．毎回全ての文章との 関連性を計算しないために，検索エンジンが単語と文章の類似度を転置ファイルに保存し，質問の単語と文章の類似度の和を質問とその文章の関連性とする．このような計算が必要であるため，[] などキーワード検索しか対応できない研究は類似検索に応用できない．

PDS をはじめに多くの曖昧化検索メカニズム [] は質問の全体を分析し, 適切な $K-1$ 個のダミー質問を選ぶ．質問の全体ではなく単語ごとにダミー単語を混ぜれば，真の質問である可能性がある質問数が増え，攻撃者が真の質問を見破る確率が下がる．質問者がいつのトピックに対して検索するとき，一つの単語を複数回使うと考えられる．毎回違うダミー単語を混ぜると同じ質問者の質問に出る頻度が高い単語が真の質問単語となる可能性が頻度が低い単語より大きくなる．そんなリスクを防ぐため ETSQ は単語バケットを事前に作り，真の質問単語と同じバケットにある他の単語をダミー単語とする．また，単語ごとダミーを混ぜるため長い質問と類似検索に対応できる．

3.2.1　類似検索

コーパス D における検索エンジンが質問を処理するとき基本的には転置ファイルを用いている．転置フィルは質問単語の集合 W と全ての単語の転置リストからなる．単語 $w_i \in W$ の転置リスト L_i が $\langle d_i, p_{ij} \rangle$ の列である．$p_{ij \in \Re}$ は単語 t_i と文¿章 $d_i \in \mathcal{D}$ の関連性である．t_i が d_i に現れたなら p_{ij} の値は 0 より大きい，現れなかったなら 0 となる．空間圧縮のために $p_{ij} = 0$ な d_i は L_i に含まれていない．

質問 $q = \{w_i\}$ と文章 d_i と関連性は以下のように計算する

$$Score_{d_j,q} = \sum_{w_i \in q} p_{ij} \tag{3.11}$$

したがって転置リスト L_i に含まれている文章だけが 0 以上のスコアを持ち，q と関連があると見なす．転置フィルを全体暗号化しても，サーバーは転置リストの長さとアクセス頻度などの情報から真の関係値を推定できるため，そのような方法は無意味だと考えられる．

3.2.2　単語バケツ

本節では単語バケツを作る方法を述べる．まずアルゴリズム 2 を用いて WordNet データベース中の意味的近い単語を隣にして全ての単語一列に並べる．リンクが多い synset が意味的に豊富であるため，単語を一列に繋がる種として使われ，synset の関係数が多い方から小さい方への順で処理する．複数の意味を持つ単語が属する synset が意味的近いと考え，同じ単語を持つ synset を隣に並べる．また反意関係，上位下位関係，全体部分関係を持つ synset を隣に並べる．2 つの操作により，列に近い単語の意味も近いと保証する．

WordNet データベースにアルゴリズム 1 を行った結果データベース中全ての 117, 798 個の名詞を一列に並べ，アルゴリズムに有効性を証明した．

3.2.3　バケツ作り

次ではアルゴリズム 2 で出力した単語列を単語バケツにする．アルゴリズム 2 がその過程を表している．バケツの大きさを 1 $\leq$BktSz$\leq$ N に設定する．バケツの数が #Bkts= N/BktSz である．同じバケツ中の単語を可能な限りに違う意味にするため

Algorithm 2 単語を一列に並べる

```
1: function ProcessSynset(synset ss)
2:     if ss の単語が複数の既存な単語列に含まれている then
3:         そんな単語列を結合する
4:         結合した単語列を sq にする
5:     else if ss の単語が既存な単語列に含まれていない then
6:         新たの単語列を作る
7:     else ss の単語の一つが一つ既存な単語列に含まれている
8:         その単語列を sq にする
9:     処理していない ss の単語を sq に加える
10:    ss の単語を処理したとマークする
11:    ss を処理したとマークする
12:    単語列 sq を返す
13: function SequenceVocab(WordNet wndb)
14:    全ての synset を関係数が多い方から小さい方への順で並べる
15:    全ての synset を処理していないとマークする
16:    全ての単語を処理していないとマークする
17:    SeqSet = φ
18:    for all 処理していない synset ss do
19:        sq = ProcessSynset(ss);sq を SeqSet に加える
20:        for all ss と反意関係，上位下位関係，全体部分関係をもつ synset ss′ do
21:            処理していない ss′ の単語を sq に加える
22:            ss′ の単語を処理したとマークする
23:            sq = ProcessSynset(ss′);sq を SeqSet に加える
```

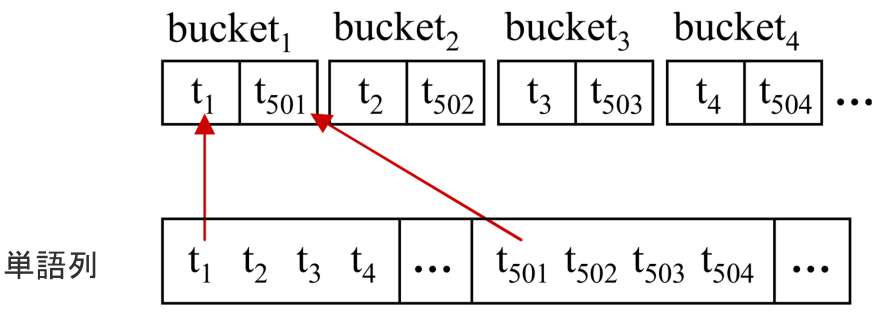


図 3.1. バケツ作り-$N = 1000, BktSz = 2$

Algorithm 3 単語列から単語バケツを作る

```
1: function GenerateBuckets(sq,BktSz,Segsz)
2:     N = 単語列 sq の長さ
3:     #Seg = N/SegSz
4:     sq を同じ長さのセグメントに分割する S_1, S_2, ..., S_{#Seg}
5:     セグメント中の単語を特殊レベルが大きい方から小さい方への順で再配列する
6:     for i = 1 to N/(BktSz * SegSz) do
7:         ActiveSeg = φ
8:         for j = 1 to BktSz do
9:             ActiveSeg = ActiveSeg ∪ S_{(j-1)N/(BktSz*SegSz)}
10:        for j = 1 to SegSz do
11:            新たなバケツ B = φ を作る
12:            ActiveSeg 中の全てのセグメントの j 番目の単語を B に入れる
13:            B を出力する
```

に単語列の $1, \#\text{BktSz}+1, 2 * \#\text{BktSz}+1, \ldots, (\text{BktSz}-1) * \#\text{BktSz}+1$ 番目の単語をバケツ 1 に，$2, \#\text{BktSz}+2, 2 * \#\text{BktSz}+2, \ldots, (\text{BktSz}-1) * \#\text{BktSz}+2$ をバケツ 2 に，$i, \#\text{BktSz}+i, 2 * \#\text{BktSz}+i, \ldots, (\text{BktSz}-1) * \#\text{BktSz}+i$ をバケツ i に入れる．図 3.1 が $N = 1000, BktSz = 2$ のときのバケツ作り過程を表している．その操作により，2 つのバケツ i と j の同じ位置の単語間の距離が同じ $\|i - j\|$ であり，意味的な距離の差も小さいと考えられる．またバケツ同じ位置の単語間の距離が違う位置の単語間の距離より近いため、真の質問の単語が同じ位置にあると仮定する。したがって，真の質問の単語が意味的に近いときあるいは一つのトピックに集中したとき，バケツの中のダミー単語も同じように一つのトピックに集中すると考えられる．しかし，バケツ中の単語の特殊レベルがランダムであり，大きく違う可能性がある．

バケツ中の単語の特殊レベルを調整するために，隣接のバケツ間の単語交換を行う．実践的には単語を単語バケツに配置する前に単語列を同じ長さ SegSz$\leq N/$BktSz のセグメントに分割し，セグメント内の単語を特殊レベルが大きい方から小さい方への順で再配列する．SegSz が BktSz の整数倍である必要がある．図 3.2 が図 3.1 の上に単語列の再配列を加えた流れを表している．その結果，同じバケツにある単語のセグメント内の順番が同一であり，特殊レベルが近くなると考えられる．

バケツ作りには 2 つのパラメータを設定する必要ある．SegSz が 2 つのリスクのトレードオフとなる．SegSz が増加することは単語交換を行う範囲が増大することに相当する．SegSz が大きければ大きいほどバケツ中の単語の特殊レベルが近くなる．一方、単語間の意味的な距離も近くなる可能性がある．もう一つのパラメータ BktSz がプライバシーと計算時間のトレードオフとなる．BktSz が大きくなると，真の質問を特定する可能性が下がるが，検索エンジンが処理する質問単語が増加する．

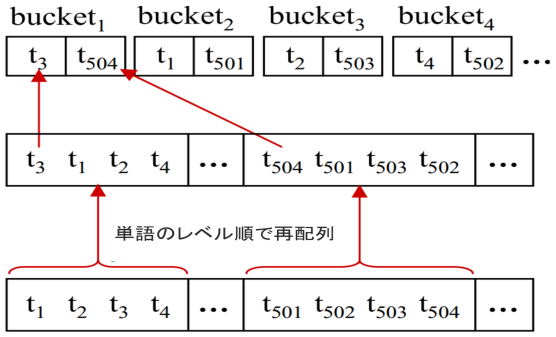


図 3.2. バケツ作り-$N = 1000, BktSz = 2, SegSz = 4$

3.2.4 プライベート検索スキーム

本節では真の質問単語だけの関連性スコアを計算できる検索スキームを述べる．検索スキームは質問加工，質問検索と結果処理三部分からなる．

Algorithm 4 質問加工

1: **function** GENERATEBUCKETS(sq,BktSz,Segsz)

Input: 真の質問単語 t_i の集合

Output: 加工した質問 q

2: **for all** 真の質問単語 t_i **do**

3: Bkt= t_i が属する単語バケツ

4: **for all** $t_j \in$Bkt **do**

5: **if** $t_i == t_j$ **then** $\mu_j = 1$

6: **else** $\mu_j = 0$

7: $E(u_j) = g^{\mu_j}\mu^r$

8: $\langle t_j, E(\mu_j)\rangle$ を q に入れる

アルゴリズム 4 が質問加工の流れを表す．真の質問単語が属するバケツの中の他の単語を全てデミー単語として質問に加える．デミーを加えた質問の単語 t_j に $E(\mu_j)$ を付け，t_j が真の質問単語なら $\mu_j = 1$，ダミー単語なら $\mu_j = 0$．$E(\cdot)$ は加算可能な準同型暗号 [?] の暗号化関数である．加算可能な準同型暗号が以下 2 つの特徴を持つ．二つの暗号文 $E(m_1), E(m_2)$ が与えられた時に，平文や秘密鍵なしで $E(m_1 + m_2)$ を計算できる．また，同じメッセージ m が複数の暗号文に対応でき，攻撃者が暗号文の頻度から m を推定することを防げる．

アルゴリズム 5 がサーバー側の検索過程を表す．サーバーが単語と文章の関連値を保存している転置フィルを用いて文章の関連性スコアを計算する．加算可能な準同型暗号の特徴より，$E(\mu_j)^{p_{ij}} = E(\mu_j * p_{ij})$．$t_j$ がダミー単語であれば，$E(score_j) * E(\mu_j)^{p_{ij}} = E(score_j) * E(0 * p_{ij}) = E(score_j)$．復号した関連性スコアには影響を与えない．したがっ

Algorithm 5 質問検索

1: **function** GENERATEBUCKETS(sq,BktSz,Segsz)

Input: 加工した質問 q

Output: 文章とその文章暗号文した関連性スコアの集合 R

2: $R = \phi$
3: **for all** $\langle t_i, E(\mu_i)\rangle \in q$ **do**
4: **for all** $\langle d_j, p_{ij}\rangle \in L_i$ **do**
5: **if** $\exists\langle d_j, E(score_j)\rangle \in R$ **then**
6: $E(score_j) = E(score_j) * E(\mu_j)^{p_{ij}}$
7: **else**
8: $\langle t_j, E(\mu_j)^{p_{ij}}\rangle$ を R に入れる

て，$score_j$ が真の質問単語と文章の関連値 p_{ij} の和となる．

最後に質問者がサーバーがらもらった結果集合の関連性スコアを復号し，その値を用いて文章を再配列するとプライバシー保護手法を使っていない検索エンジンと同様な検索結果がもらえる．

3.2.5　プライバシー分析

3.3　質問意図を曖昧化するキーワード検索

HDGA は [] 提案した潜在的ディリクレ配分法 (LDA) に基づく質問意図の曖昧化メカニズム (TIO) の改良手法である．LDA の詳細は第 4 章で述べる．HDGA が以下の特徴を持つ，まず，サーバーに提出した質問グループに属する各質問が違うトピックに属し，ダミー質問の生成過程が相互独立である．

次に，HDGA は TIO のように真の質問をカバーできるトピックからダミー質問を作るではなく同じ質問グループに属する質問が同じ地位を持つ．

そして，HDGA がハッシュ関数 Highest Random Weigh(HRW)[] を用いてダミートピックを選び，トピックの出現頻度を均一にする．

アルゴリズム 6 が HDGA の質問生成メカニズムを表す．ここで $Pt[w|t]$ が LDA 分析の結果であり，h が HRW ハッシュ関数である．

3.3.1　プライバシー分析

Algorithm 6 HDGA(On Masking Topical Intent in Keyword Search)

Input: 質問:q_1

1: $Q = \{q_1\}\delta_{q_1} = \underset{t \in T}{\operatorname{argmax}}\, Pr[t|q_1]$
2: **for all** $t \in T \setminus \{\delta_{q_1}\}$ **do**
3: $\quad e_t = h(\delta_{q_1}||t||s)$
4: $T_D = \{t^1_{q1}, t^2_{q1}, \ldots, t^2_{q1} | \forall t_1 \ \in T_D, \forall t_2 \ \in T \setminus T_D, e_{t_1} > e_{t_2}\}$
5: **for all** $t \in T_D$ **do**
6: $\quad$ **while** $\underset{t \in T}{\operatorname{argmax}}\, Pr[t|q'] \neq t$ **do**
7: $\quad\quad Pr[w|t]$ に基づいて $|q_1|$ 個の単語をランダムに選び，ダミー質問 q' を作る
8: $\quad Q = Q \cup \{q'\}$
9: Q をシャッフルする

Output: Q

第 4 章

意味分析

4.1 tf-idf

4.2 **潜在意味解析**

4.3 **潜在的ディリクレ配分法**

第 5 章

プライバシー分析 (攻撃手法)

本論文では攻撃者が質問者が質問意図を隠していることと質問者が用いている質問曖昧化手法のメカニズムを知っているを前提とし，攻撃手法を考える．

曖昧化検索は 3 つの違うレベルな目標があると同じように曖昧化検索に対する攻撃手法も 3 つの違うレベルな目標がある．ダミー質問が混ぜられた質問グループから真の質問 q_R を見つかる．ダミー質問が混ぜられた質問グループから質問者が検索したいものを見つかる．ダミー質問が混ぜられた質問ログから質問者が興味を持つトピックを見つかる．

1 つ目の目標に対して本論文では質問 q と質問 q のメイントピック δ_{SA} 間の関連性を攻撃するメイントピック攻撃を提案する．質問者が検索したいものを定義するのは難しいため，本論文ではダミー質問の検索結果と真の質問の検索結果が一致する割合を用いて評価する．そして，3 つ目の目標を達成できる既存な攻撃手法類似度攻撃 [4] を紹介し，改良手法を提案する．

5.1 メイントピック攻撃

ダミー単語が真の質問単語と同様にいつのトピックに集中することが失敗したら，真の質問のメイントピックと関係が強い単語が他のトピックと関係が強い単語の数より多い，加工した質問のトピックと真の質問のトピックが一致することが考えられる．また，一つのバケツの中の単語が意味的に遠いため，ダミー単語が真の質問単語のメイントピックとの関連性が弱いと考えられる．メイントピック攻撃では各単語バケツ中質問のメイントピックと一番関連性が強い単語を真の質問の単語と推定する．アルゴリズム 7 はその流れを表している．

5.2 類似度攻撃 [4](事前情報あり)

5.3 類似度攻撃 2(事前情報なし)

Algorithm 7 メイントピック攻撃

Input: 質問:$q = \{t_i\}$, 単語のトピックベクトル集合 $L = \{\ell_i\}$

$R = \phi$, $\ell = 0$

$\ell = \sum_{t_i \in Q} \ell_{t_i}$

$maintopic = \underset{j}{\operatorname{argmax}}\, \ell[j]$

for all $bk_k \in q$ **do**

　　$R = R \cup \{\max_{t_i} l_{t_i}[maintopic]\}$

return R

Algorithm 8 類似度計算

Input: 質問 q, ユーザープロフィール P_u, スムージングパラメータ:α

for $q_i \in P_u$ **do**

　　$coef[i] \leftarrow 2 \cdot |q \cap q_i| \cdot \frac{1}{|q|+|q_i|}$

$coef \leftarrow sort(coef)$

$sim \leftarrow coef[0]$

for $i \in [1, |P_u|]$ **do**

　　$sim \leftarrow \alpha \cdot coef[i] + (1 - \alpha) \cdot sim$

Output: sim

Algorithm 9 類似度攻撃

Input: 質問集合 Q, ユーザープロフィール Pu, スムージングパラメータ:α

$q^* = \underset{q \in Q}{\operatorname{argmax}}\, sim_{q,Pu}$

Output: q^*

Algorithm 10 類似度攻撃

Input: 質問集合列 $\hat{Q} = \{Q_1, Q_2, \ldots, Q_n\}$, スムージングパラメータ:$\alpha$

for $j \in |Q_1|$ **do**
  $\hat{Pu}[j] = Q_1[j]$
  $\hat{Put}[j] = \Phi$
  $d[j] = 0$
for $i \in [2, n]$ **do**
  for $j \in |Q_i|$ **do**
    $\hat{Put}[j] = \underset{Pu \in \hat{Put}}{\operatorname{argmax}}\, sim_{Q_i[j], \hat{Put}[j]}$
  $q_i^* = \underset{Q_i[j] \in Q_i}{\operatorname{argmin}}\, sim_{Q_i[j], \hat{Put}[j]}$
  for $j \in |Q_i|$ **do**
    $\hat{Pu}[j] = \hat{Put}[j] \cap Q_i[j]$

Output: q^*

第 6 章

質問曖昧化 (提案手法)

事前に単語をグループにする [2]，質問をグループにする [1] と同じようにトピックをグループにすることよりトピック出現頻度で質問者が興味あるトピックを特定することが防ぐと考えられる．

[3] ではハッシュ関数でトピックをグループにしているが，各トピック間の関係を配慮していない．また，[3] では各ダミートピックから単語をランダムに選ぶため，真の質問に含まれている単語は違っても属するトピックは同じならダミー質問が同じような性質を持つ。一方，同じ真の質問に対して同じダミー質問を生成することができない．真の質問に含まれている単語という情報を用いてないことが [3] に提案した手法が Simattack に弱い原因だと考えられる．simattack から真の質問を守るために真の質問が同じ単語を含むとき，ダミー質問も同様に同じ単語を含んでほしい．それを実現するため提案手法では単語ベクトルを用いた．

6.1 単語ベクトル

定義 8. 単語ベクトル T を全てのトピックの集合とし W を全て単語の集合とする．トピック t の単語ベクトル l_t とは

$$
\begin{aligned}
&l_t = \{w_1, w_2, \ldots, w_{|W|}\}, \\
&\forall w \in l_t, w \in W \\
&\forall 1 \leq i \neq j \leq, w_i \neq w_j \\
&\forall 1 \leq i < j \leq |W|, rscore(w_i, t) \geq rscore(w_j, t)
\end{aligned}
\tag{6.1}
$$

質問のメイントピックを計算し，質問に含まれている単語をその単語が質問のメイントピックの単語ベクトルにいる順番にすれば，質問を数字ベクトルで表わすことができる．同様にトピックが決めれば，そのトピックの単語ベクトルを用いて数字ベクトルを質問に翻訳することができる．

単語ベクトル内の単語が単語とそのトピックの関連値の大きい方から小さい方までに並ぶため，単語ベクトルに同じ順番を持つ単語がその単語ベクトルを持つトピックに対して同じ様な関連性を持つと考えられる．また，同じ数字ベクトルで表わせる質問もその質問が属するトピックに対して同じ様な関連性を持つと考えられる．

したがって，単語ベクトルを通じて違うトピックに属するが似たような特徴を持つ質問を作るこたができる．

6.2 質問曖昧化

第 7 章

データベース分割

特許分類を用いることにより特許データベースを分割することができる．分割したデータベース各々に対して同じような信憑性を持つ質問を提出すると真に検索したいデータベースを隠すことができると考えられる．

第 8 章

評価実験

重複を除いた単語数	2,973,096
文章数	3,496,253
質問数	2,908
質問平均単語数	21.0
国際特許分類数	623

表 8.1. データベース

8.1 tfidf vs lda vs lsa

評価方法:ダミー質問数	3	4	5	6	7
SimAtt New		78.2	78.5	77.3	63.8
SimAtt LSA	65.5	55.6	50.8	48.2	50.7
Maintopic	75.2	72.3	67.6	64.2	58.5
100 番までの検索結果重複率	6.0	1.9	2.0	1.9	1.9

表 8.2. SA:LSA ダミートピック選び方:ランダム ダミー質問数の影響 1

評価方法:ダミー質問数	3	4	5	6	7
SimAtt New		82.8	77.0	74.4	72.6
SimAtt LSA		52.5	47.8	43.7	42.4
Maintopic		81.8	78.0	76.0	74.1
100 番までの検索結果重複率		1.5	1.9	1.7	1.6

表 8.3. SA:tfidf ダミートピック選び方:ランダム ダミー質問数の影響 2

8.2　データベース分割

評価方法:ダミトピックの選び方	cos-near	cos-far	coef-near	coef-far
SimAtt New	72.7	86.0	66.8	90.3
SimAtt LSA	33.5	51.8	34.5	87.2
Maintopic	38.3	77.5	28.2	69.2
100 番までの検索結果重複率	2.8	2.0	2.8	1.7

表 8.4. SA:LSA ダミー質問数:3 ダミトピックの選び方の影響 1

評価方法:ダミトピックの選び方	coef-near	coef-far
SimAtt New	76.3	86.9
SimAtt LSA	63.4	63.1
Maintopic	77.7	87.4
100 番までの検索結果重複率	6.5	1.3

表 8.5. SA:tfidf ダミー質問数:3 ダミトピックの選び方の影響 2

評価方法	SimAtt New	SimAtt LSA	Maintopic	重複率
tfidf	25.1	25.0	25.1	28.5
LSA	27.6	27.0	25.8	20.7
LDA	25.8	24.6	24.8	32.3

表 8.6. ダミー質問数:3 ダミトピックの選び方:真の質問と同じトピック SA の影響

評価方法	SimAtt New	SimAtt LSA	Maintopic	重複率
データベース分割 (8)	52.3	11.6	42.1	1.5

表 8.7. データベース分割

第 9 章

おわりに

謝辞

参考文献

[1] “Providing Privacy through Plausibly Deniable Search”, Proceedings of the 2009 SIAM International Conference on Data Mining, Proceedings, Society for Industrial and Applied Mathematics, pp. 768–779 (2009).

[2] “Embellishing Text Search Queries to Protect User Privacy”, Proc. VLDB Endow., **3**, 1-2, pp. 598–607 (2010).

[3] “On masking topical intent in keyword search”, 2014 IEEE 30th International Conference on Data Engineering, IEEE, pp. 256–267 (2014).

[4] “SimAttack: private web search under fire”, Journal of Internet Services and Applications, **7**, 1, p. 1 (2016).

28

付録 A