

機能に基づいたアプリ検索手法の提案

上西達也

背景、目的

新規アプリ普及の課題

- スマートフォンなどのアプリケーション(アプリ)は大量に出回っており、新規アプリも日々開発、リリースされている
- ユーザはそこから自由に選んで端末にインストールするが、**すでに人気のアプリを採用するユーザ**もいれば、**自ら好みのアプリを探して採用するユーザ**もいる
- 新規アプリを普及させたいときは、まず **自らアプリを探すユーザ**にアプリを見つけてもらう必要がある

自らアプリを探すユーザ

Everett M. Rogersは新しいアイデアや技術の採用が早い順にユーザを5つに分類した(イノベーション普及学)

1. イノベーター : 冒険心にあふれ、新しいものを進んで採用する(2.5%)
2. アーリーアダプター : 流行に敏感で、自ら情報収集する(13.5%)
3. アーリーマジョリティ : 新しいものを取り入れるが比較的慎重(34.0%)
4. レイトマジョリティ : 周囲の大多数を見てから同じ選択をする(34.0%)
5. ラガード : 流行や世の中の動きに関心が薄く、最も保守的(16.0%)

イノベーターの特徴

- 新しいければとりあえず採用してみる姿勢、冒険心がある
- 後に普及しないアイデアを採用することもある(リスク許容度が高い)
- 新着アプリ一覧などを用意するだけで次々に採用してくれる可能性があると考えられる
- 今回の研究では、イノベーターは対象から外す

アーリーアダプターの特徴

- 自らの利益を考慮しながら採用する(リスク許容度が低い)
- アプリ購入前にその機能を把握し、求めている機能を持ったアプリを選びたい
欲求があると考えられる

【最新】シンプルエディタ



【人気】高機能エディタ



無駄な機能に興味がない
アーリーアダプター



現在のアプリ検索システムの問題点

1. すでに人気のあるアプリを上位に表示する傾向がある※1
 - ダウンロード数が少ない**新規のアプリ**などはたとえ優れていてもアーリーアダプターの目に触れにくい
2. タイトルや説明文などの質によって順位が大きく変わる
 - アプリストア最適化 (ASO) が順位操作に効果を上げている現状がある※1
 - 全く機能の違う偽アプリが上位になることも※2

※1 : App Annie (www.appannie.com)

※2 : 「偽アプリに気を付けて: ポケモンGO便乗不正アプリが3900万件ダウンロード、Google Playでの配布も - ITmedia PC USER」(www.itmedia.co.jp/pcuser/articles/1608/18/news091.html)

アプリの機能に基づく検索の必要性

- アーリーアダプターは人気よりも求めている機能があるか（利益があるか）に着目する
- 純粋に機能に基づいたアプリ検索システムがあれば、アーリーアダプターが求めている機能を持ったアプリを見つけやすくなる
- 新規でもよりよいアプリが普及し、ソフトウェア業界の活性化に繋がる

対象とするユースケース

既知のアプリと機能面で類似するアプリを検索する場合

- 欲しい機能があるもの
- いない機能がないもの

アプローチ

アプリ解析における課題

- アプリの多くはソースコードが公開されていない
- 従来からソフトウェア検索手法に関する研究はあったが、ソースコードが手に入ることが前提となっている
- 既存手法は使えない
- 実行可能ファイルなどから抽出できる情報のみを使った手法が必要

ソフトウェアバースマーク

※birthmark: (生まれつきの)あざ、ほくろ

個々のソフトウェアが元来持っている**固有の特徴の集合**のこと

(例) 初期値代入、使用クラス、メソッド呼び出し列、継承関係

- ソフトウェアの盗用を発見、証明するための研究にて定義されている
- **ソースコードが手に入らなくても取得できる特徴**で定義している

Stigmata

- Javaのクラスファイルからソフトウェアバースマークを抽出するために開発されたツール
- JARファイルから標準API(メソッド)呼び出し列などを抽出できる
 - 多くの難読化ツールが標準API呼び出しまでは難読化しないことを利用している

アプリ解析のアイデア

一般的なアプリ開発では、標準APIを組み合わせて目的の機能を実装する

↓ だから

各クラス内の標準API呼び出しの組み合わせの類似性(共起関係の類似性)は
機能と関連があると考えられる

↓ そこで

StigmataがJARファイルから抽出するソフトウェアバースマーク(標準API呼び出し)のデータを応用する

解析結果の利用方法

Stigmataの標準API呼び出し列バースマークは呼び出し元クラスごとにまとめて出力される

- 標準API呼び出し列の順序に機能的意味を見出だせない状態



クラスを文書、標準API呼び出しを単語に対応させて共起行列 (Bag of Words モデル)を作成して利用する

類似アプリ検索手順(v0.1)

1. 各APKファイルから、クラスごとの標準API呼び出しの組を抽出
2. クラスと標準API呼び出しでBag of Words共起行列を作成
3. 潜在的意味インデキシング(LSI)を適用
4. 既知のアプリ(検索の入力)の各クラスと同じ機能を持つと推定されるクラスをより多く持つアプリを提示

類似アプリ検索手順(v0.2)

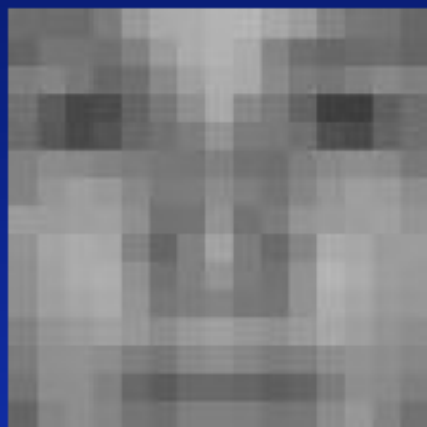
1. 各APKファイルから、クラスごとの標準API呼び出しの組を抽出
2. クラスと標準API呼び出しでBag of Words共起行列を作成
3. クラスがベクトルであることからアプリをベクトル部分空間とし、相互部分空間法を適用

相互部分空間法

- 2つのベクトル部分空間の類似度を計算する方法
- それぞれの部分空間の正規直交基底ベクトルがなす正準角 $\theta_1 \sim \theta_n$ (θ_i : i 番目に小さい正準角)のコサインの2乗の平均値を、部分空間間の類似度とする
- 画像処理のパターン認識でよく使われている
- 特異値分解を用いて簡単に実装できる

画像パターンのベクトル表現

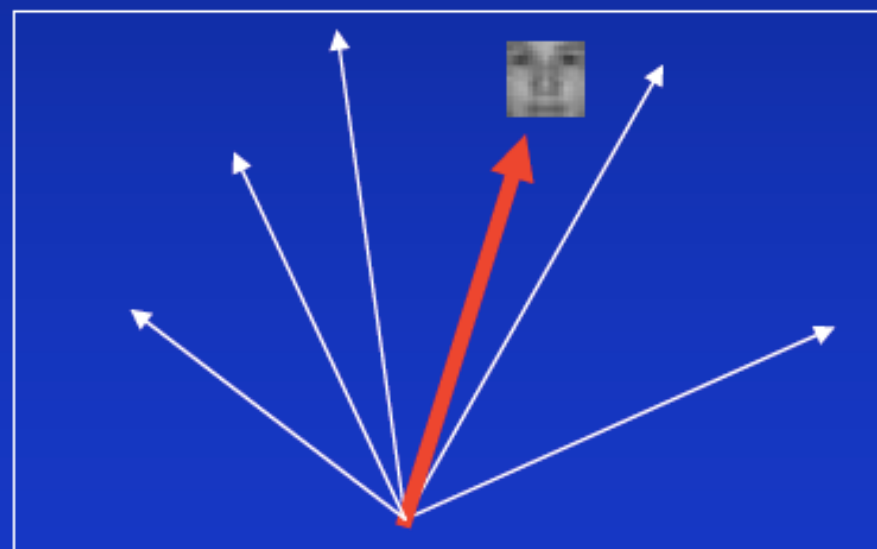
K画素



K画素



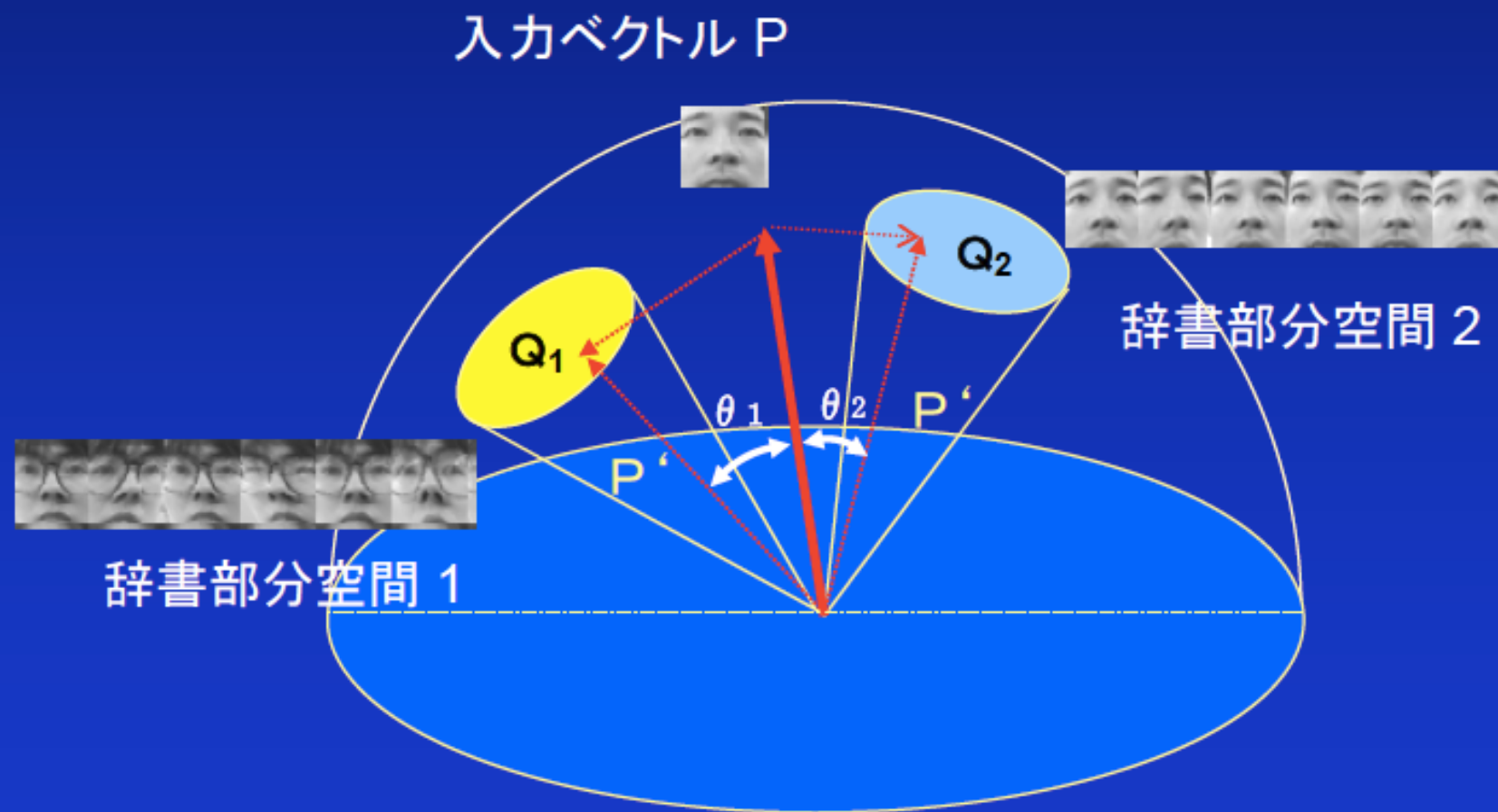
$K \times K$ 次元のベクトル
この例では225次元



$K \times K$ 次元 空間

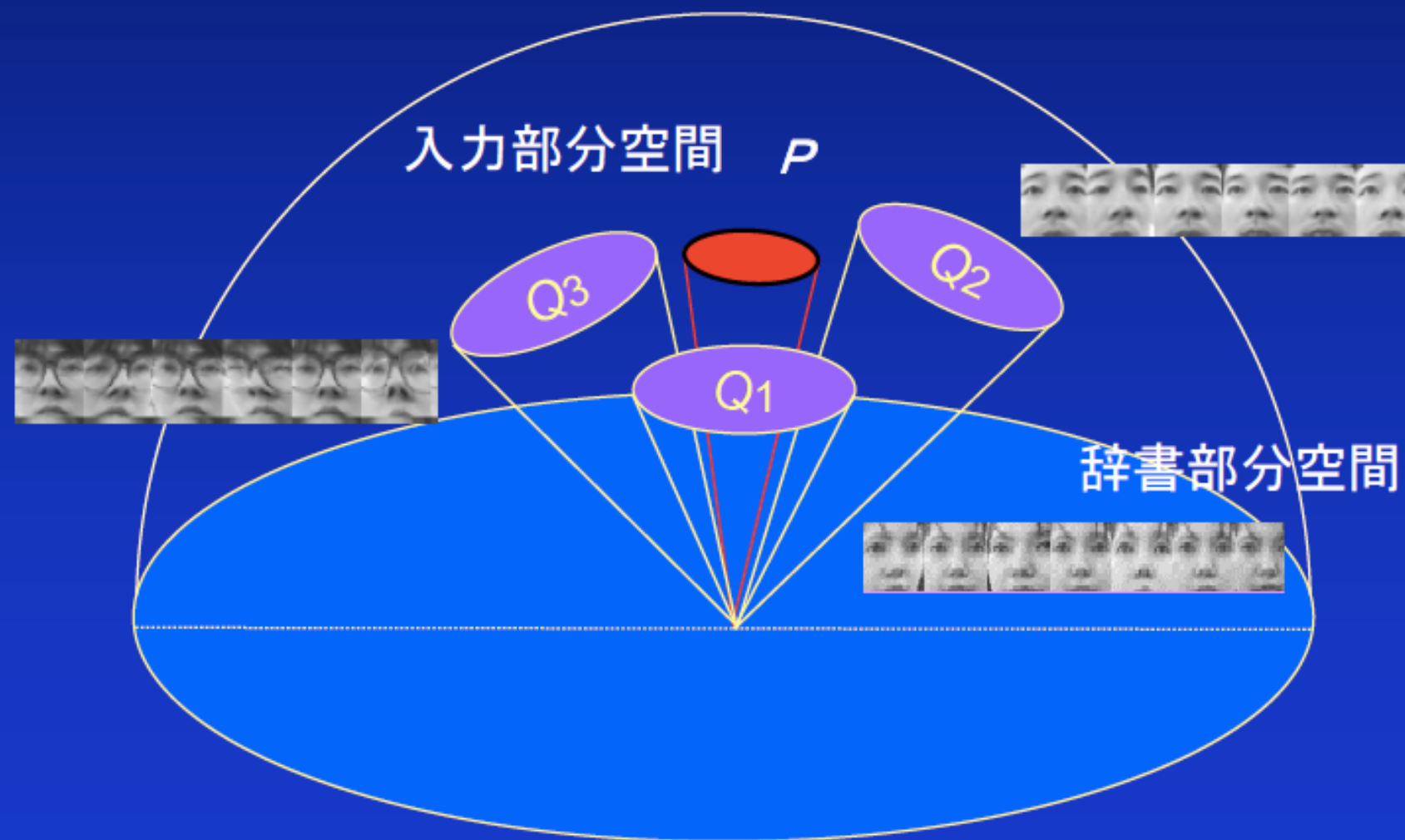
部分空間法（複合類似度法）

入力ベクトル P と辞書部分空間 Q の角度 θ (射影長 $|P'|$) を類似度として識別する

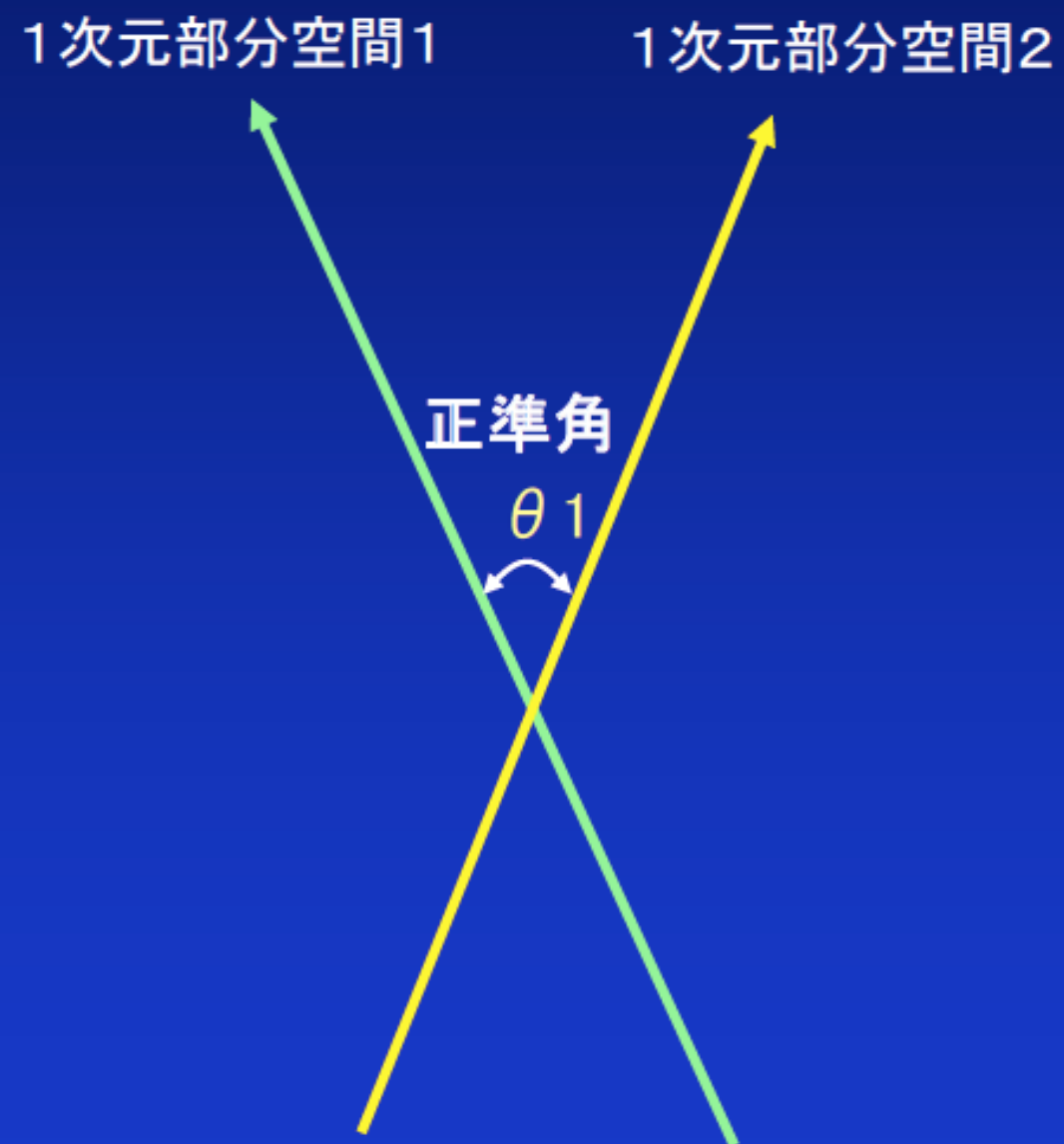


相互部分空間法

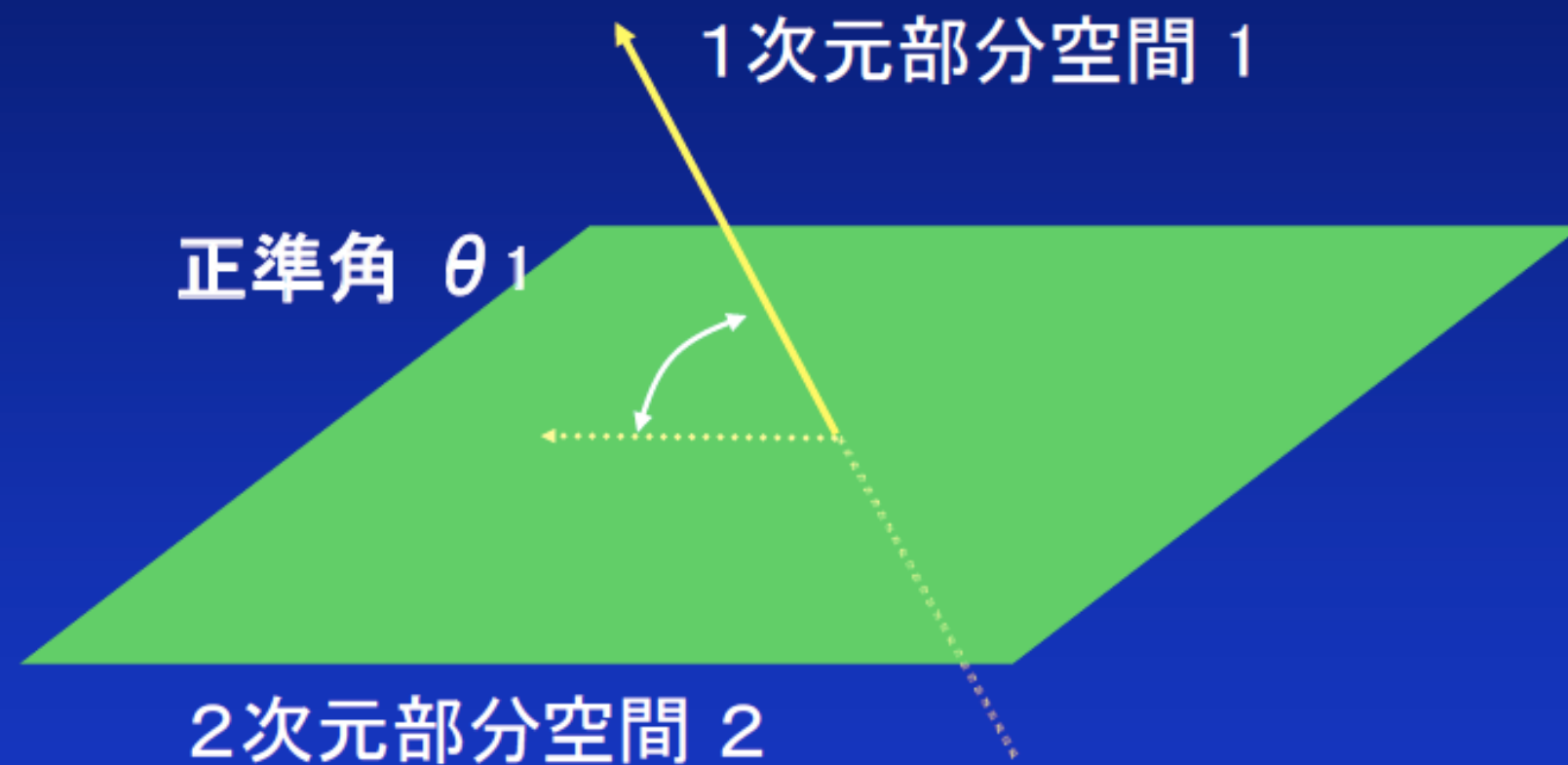
入力部分空間 P と辞書部分空間 Q のなす正準角度 θ に
基づいて識別する（前田，渡辺，1985）



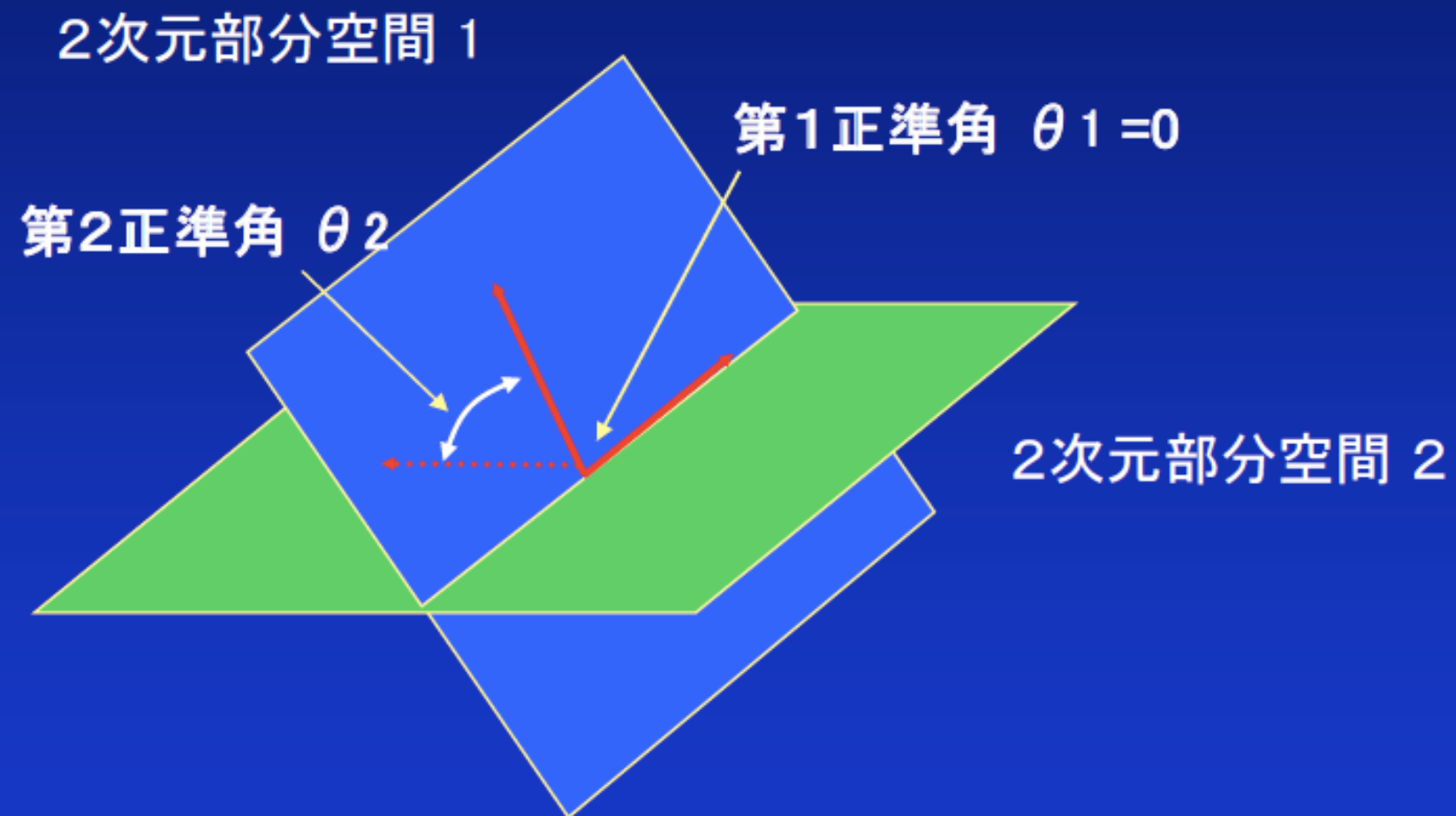
正準角の例 1



正準角の例 2



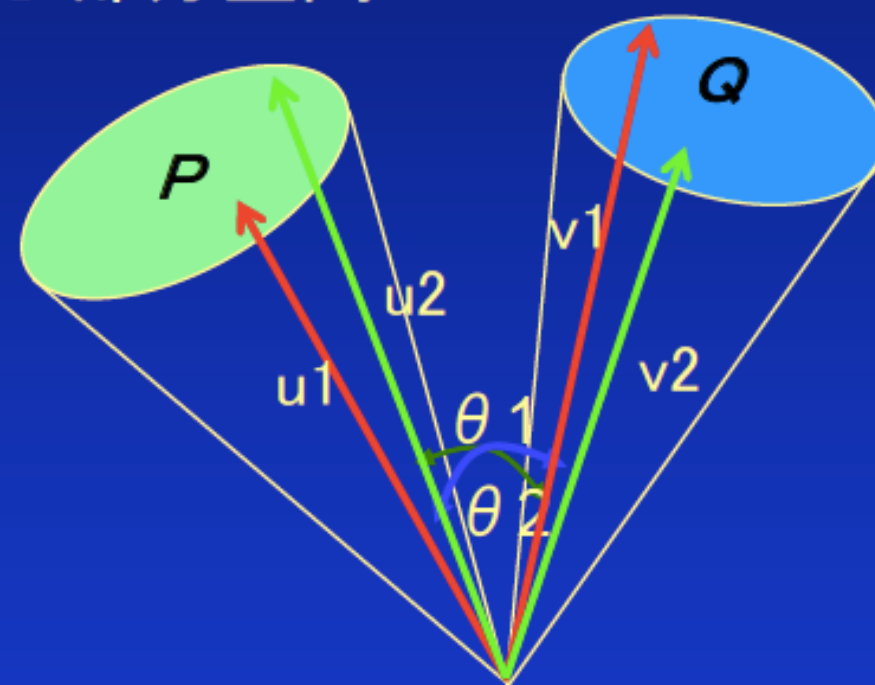
正準角の例 3



正準角の例 4

K 次元 部分空間

K 次元 部分空間



K 個の正準角 θ_i ($i=1-k$)

差分抽出について

- 差分は特定の2アプリ間において定義される
- クラス間類似度がより小さいものを提示すればいいのでは？
- (メソッド数の少ないクラスの扱いが重要になりそう)

アプローチまとめ

- Stigmataを用いてJARファイルからクラスごとの標準API呼び出しの組み合わせを抽出する
- クラスを文書、標準API呼び出しを単語に対応させて共起行列化する
- 潜在的意味インデキシング(LSI)を用いて、クラス間の類似度を計算し、同じ機能を持つと推定されるクラスをより多く持つアプリを提示するか、もしくはクラスがベクトルであることからアプリをベクトル部分空間とし、相互部分空間法を適用

ケーススタディ

検索対象

- Androidアプリを対象とする
 - 実行可能ファイルはAPKファイル(JARファイルに変換できる)

現在の気付き

- 名前に\$マークが付いた謎クラスがいくつも見つかった
 - クラスの中にクラスが宣言できるらしい(内部クラスと呼ばれるらしい)
 - またenum型は内部的には内部クラスとして表現されているらしい
 - 内部クラスはコンパイルすると名前に\$マークが付いた別ファイルとして生成されるらしい
 - これらの仕業だったらしい
- 標準APIを含むほぼ全てが難読化されているアプリを発見した(解析できない)