文書分類手法を利用した
画像検索結果のフィルタリング

正木裕一
茨城大学大学院
理工学研究科

平成18年2月10日

概要

本論文では、Google の画像検索結果に対し、文書分類の手法を利用したフィルタリングを行う。これによって、画像検索の精度の向上、目的の画像に達するまでの時間を短縮化することができる。

現在、一般的な通信回線の高速化もあり、ネットワークを介して、画像などの容量の大きなマルチメディア情報を自由にやりとりできるようになった。それにともない、マルチメディアを検索するマルチメディア検索技術が要求されている。特に画像検索は、利用する目的が医療の分野から、趣味の分野まで多岐に渡り、検索精度の向上、目的の画像に達するまでの過程の簡略化が求められている。

現在、Google は「Google 画像検索」という画像検索サービスを提供している。画像の検索方法は、通常の検索と同様、検索窓に検索キーワードを入力することで行う。Google 画像検索では画像周辺のテキスト、画像のキャプション、およびその他数多くの要因を分析することで、検索キーワードとのマッチングが最も高い画像をトップにランキングする。つまり、その実体はテキスト分析による画像の索引化であり、画像そのものではなく HTML 文書中のメタ情報や、その他の文章といったものが検索されている。そのため、現状の画像検索では、検索キーワードの要求と関係の無い画像が上位にランキングされる場合も多々ある。

このため近年、HTML テキストの重要文を用いた画像ラベリング手法なども提案されているが、未だに高性能な画像検索システムは提供されるまでにいたっていない。そこで、本論文では「Google 画像検索」の検索結果をフィルタリングして、要求に合わない画像を取り除いたり、検索結果のランキングを変更したりする。これによって、画像検索の精度向上を目指す。また画像検索作業による負荷を軽減することもはかる。具体的には、検索された画像を含む HTML 文書のテキスト部分の内容と、入力されたキーワードとの分野的な関連性が高いかどうかに注目する。検索キーワードとその検索された Web ページのテキストの内容を比較し、検索キーワードと文章の内容がかけ離れている場合、その Web ページの画像を排除、もしくは重みを低くすることによって画像検索の精度の向上をはかる。また同時に、検索過程の簡略化も実現する。

ここでは検索キーワードと、画像を含む HTML 文書の属している分野の関連性が高いかどうかを調べるために、Naive Bayes 法を利用した。Naive Bayes 法は確率論に基礎をおく分類手法の 1 つである。実験では、この Naive Bayes 法を利用して Naive Bayes 分類器を作成した。また、検索キーワードごとに分類先となるクラスを作成した。分類器は、分類するための事前知識となる訓練データによって学

習を行い、分類対象となる実験データを分類をする。この分類器に実験データを入力すると、そのデータが各クラスに属しているスコアが算出される。次にそれらのスコアの総和を求め、正解となるクラスのスコアの比を算出することによって、検索されたWebページとキーワードの分野の関連性を判断する。スコア比の低いものを文書内容とキーワードの関連性が低い画像とした。また今回は対象となる画像の種類を限定し、画像の中でも特に需要が高いと思われる、人名による画像検索に特化して実験を行った。

今回は実験データとなる画像を含むWebページのテキストから抽出された単語に対し、単語の出現したWebページ内の個所による重み付けをせずに行った実験、訓練データから検索キーワードの単語を除いた実験、ALT属性に対し重み付けをして行った実験の三種類の画像検索結果のフィルタリングの実験を行った。

Webページ内の個所による重み付けをせずに行った実験では、各60件の実験データに対し実験を行ったが、同姓同名の人物の写真が省かれないなどの問題が残った。ALT属性に対し重み付けを行った実験では 独自に各20件の実験データを作成した。しかし、あまり良い結果を得ることは出来なかった。最後に訓練データから検索キーワードを取り除いた実験では、ALT属性に重み付けをした実験と同じ実験データを用いたが、ALT属性に重み付けをした実験時よりも関連性の高い画像に対し高いスコアをつけることが出来た。

今回の実験の結果、ある程度良好な結果を得ることが出来たが、関連性の無い画像を省くことができない場合もあった。しかし、Googleの順位で下位のほうにあった関連性が高い画像を上位することができるなど、一部の結果においては提案した手法の有用性を確認できた。今後の課題として、今回のNaive Bayes法を用いたフィルタリングに加え、新たな判断規準を設けることがあげられる。現在、その基準1つとして、人名の共起が多いWebページの重み付けを下げることを考えている。

Naive Bayes 法の有効性は十分に立証できたと言えるか？

abstract

In this paper, I use the method of document classification to filter the results of Google image search. This filtering system improves the precision of the image search, and reduces time to find the target image.

At present, we can use high-speed communication circuits to connect to the Internet, and load large volumes of the multimedia data like images easily. Under such environment, the multimedia search technology is required. Especially, the image search is actively studied to improve the precision of the search and to reduce the time to find the target image, because the image search is used in various domains, for example, medical services and private use.

In this service, Google provides the image search service called "the Google image search." we can search an image by using the same method to the text search, that is, inputting a search keyword into the search window. The Google image search highly ranks the image, which is most relevant to the search keyword, by analyzing various features, which are text around the image, the caption of the image and so on. That is to say, the mechanism of the image search is the indexing of text. The image search engine uses META tags of the HTML documents and its text to search the image, not the image data. As the result, even a state of the art image search ranks highly non-relevant images sometimes.

To improve it, the method using key sentences in HTML text was proposed, but it has not yet realized actually. Thus, this paper filters images retrieved by the Google image search to remove non-relevant images and to change the search ranking. This filter improves the precision of the image search. At the same time, the work operation of the image search becomes easy. Concretely, I use association between the search keyword and the text of the HTML document including the retrieved image. If the degree of association is low, the corresponding image is removed from the result of the Google image search, or is ranked lower. As the result, the precision of the search is improved. At the same time, the work operation of the image search is simplified.

To measure the degree of association between the search keyword and the text of the HTML document, I use the Naive Bayes method. The Naive Bayes method is a machine learning method based on the probability theory. In experiments, the Naive Bayes classifier is learned by this method, and the class is defined for

each search keyword. The classifier is learned from the training data, which is the pre knowledge to classify the Web page, and is used to classify the test data in experiments. Given the test data, this classifier produces each score for every classes. The score means the degree that data belongs to the class. Next, the score ratio for each class is calculated. By the score ratio, the degree of association between the search keyword and the retrieved Web page is measured. If the score ratio is low, the degree of association between the search keyword and the retrieved Web page is defined to be low. In my experiments, I limit the category of the image to the person, because the person image is much in demand.

I conduct three kinds of experiments; 1) No alternation from my original method, 2) Removal of the search keyword from the Web page, 3) Weight on words in the ALT tag.

I tried the first type approach for about 60 test data. As the result, it is shown that this approach has the problem which cannot remove images with same person name. I tried the third type approach for each 20 test data made by hand. However, this approach did not give good result. Last I tried the second type approach. In this experiment, I used the same test data to the third type approach. As a result, this approach gave higher scores than the third type approach.

These experiments gave not so bad scores, but some non-relevant images could not be removed. However, my method is useful as a whole. For examples, some low ranked images by the Google image search, but are relevant, are re-ranked highly by my method. In future, I will make new decision criterion, in addition to the filter using Naive Bayes method. Now, I am investigating the method to give low weight to the Web page, which includes many person names.

目 次

図目次

表目次

第1章　序論

1.1　背景と目的

本論文では、Google の画像検索に対し、文書分類の手法を利用したフィルタリングを行う手法を提案する。これによって、画像検索の精度の向上、目的の画像に達するまでの時間を短縮化することができる。

現在、一般的な通信回線の高速化もあり、ネットワークを介して、画像などの容量の大きなマルチメディア情報を自由にやりとりできるようになった。それにともない、Web ページからテキスト文書だけでなく、画像、動画、音といったマルチメディアデータを入手することが可能となった。しかし、Web 上にはそれらマルチメディアデータが無数に存在しているため、それら無数に存在するマルチメディアデータの中から目的とするデータを検索する、マルチメディア検索技術が要求されている。実際に、Yahoo!や Google などの大手検索サイトにおいて、検索窓に検索キーワードを入力することによるマルチメディアデータの検索が行われている。このようなキーワードによるマルチメディアデータの検索手法は、クロスメディア検索と呼ばれている。マルチメディアデータ検索の中でも、特に画像の検索は、利用する目的が CT スキャン画像などの医療の分野から、風景画像などの趣味の分野まで多岐に渡り、検索精度の向上、目的の画像に達するまでの過程の簡略化が求められている。

現在行われている標準的な画像検索では、画像周辺にあるテキストを利用している。もし画像の周辺のテキストに検索キーワードが含まれていれば、その画像を検索キーワードにマッチした画像としている。しかし、その検索結果には、画像自体を解析しているわけではないので、要求に合わない画像が検索される場合も多い。一方、画像処理を行い、画像全体の色合い、色の分布などの画像の特徴となる量を抽出し、それらの近似によって画像を検索する類似画像検索も存在する。しかし、画像インデックス作成の際に処理の負荷の高い特徴量の抽出を行う必要があり、更に検索キーワードとなる画像自体も画像処理を行う必要がある。そのため、現在広く普及しているパーソナルコンピュータの性能、通信回線速度では実現困難である。また近年、HTML テキストの重要文を用いた画像ラベリング手法 [3] なども提案されているが、今だ高性能な画像検索システムは提供されるまでにいたっていない。

そこで、本論文では Google 検索エンジンによる検索結果をフィルタリングして、要求に合わない画像を取り除いたり、検索結果のランキングを変更することによっ

て誤った検索結果を排除する。これによって、関連性が低いWebページの装飾画像などの誤って検索された画像を排除することができ、検索結果を自動的に絞り込むことによる画像検索の精度向上、またユーザの検索する画像を減らすことによる画像検索作業による負荷の軽減をすることを計る。

具体的には、検索された画像を含むHTML文書のテキスト部分の内容と、入力されたキーワードとの分野的な関連性が高いかどうかに注目する。例えば、サッカー日本代表の画像を検索するため、サッカー日本代表というキーワードを検索キーワードとして与えた場合、検索された画像の周辺のテキストは、少なからずサッカーに関連するテキストであり、まったく関係の無い文章になっていることはない。このような背景から、検索キーワードとその検索されたWebページのテキストの内容を比較し、検索キーワードと文章の内容がかけ離れているWebページの画像を排除、もしくは重みを低くすることによって検索精度の向上をはかる。また同時に、検索過程の簡略化も実現する。

今回検索キーワードと画像を含むHTML文書の内容を比較するために、Naive Bayes法 [2] を利用した。Naive Bayes法とはスパムメールフィルタリングなど、文書分類の分野において広く用いられている確率論に基礎をおく分類手法である。その特徴としては、最終的な仮定成立の確率計算に事前知識を利用し、仮定が成立するかどうかを真か偽ではなく、確率で明示的に表現できる点にある。実際の実験では、このNaive Bayes法を利用してNaive Bayes分類器を作成した。分類器は、分類するための事前知識となる訓練データによって学習を行い、分類対象となる実験データを学習した訓練データによる知識に用い分類をする。この分類器に実験データを入力すると、そのデータが各クラスに属しているスコアが算出される。本論文では、このNaive Bayes分類器に実験データを入力することによって、各クラスに属するスコアを算出する。次にそれらのスコアの総和を求め、正解となるクラスのスコアの比を算出することによって、検索されたWebページとキーワードの分野の近似性を算出することを考えた。また今回は対象となる画像の種類を限定し、画像の中でも特に需要が高いと思われる、人名による画像検索に特化して実験を行った。

1.2 本論文の構成

第2章では画像検索について、第3章では画像検索結果のフィルタリングについて説明する。第4章では今回行った文書分類を利用した画像検索結果のフィルタリングの3つの実験を報告する。4.2節では実験データとなる画像を含むWebページのテキストから抽出された単語に対し、単語の出現したWebページ内の個所による重み付けをせずに行った実験を報告する。4.3節では訓練データから検索キーワードの単語を除いた実験を報告する。4.4節ではALT属性に対し、重み付けをして行った実験を説明する。第5章では実験結果をふまえての考察を行う。第6章

では結論と今後の課題について述べる。

第2章　画像検索

2.1　Google画像検索

現在、Googleは、「Google画像検索」という画像検索サービスを提供している。Googleは4億2500万以上の画像を保存し、それらにインデックスを与えておき、検索はそのデータベースを使って行われる。画像の検索方法は、通常の検索と同様、検索窓に検索キーワードを入力することで行う。検索結果のページには画像のサムネイル表示と共に、画像の容量、サイズ、拡張子、キーワードと一致した文、サイトのURLが並ぶ。いずれかを選択すると、大きな写真と、その画像を含むウェブページが表示される。Googleイメージ検索の例を図2.1に示す。

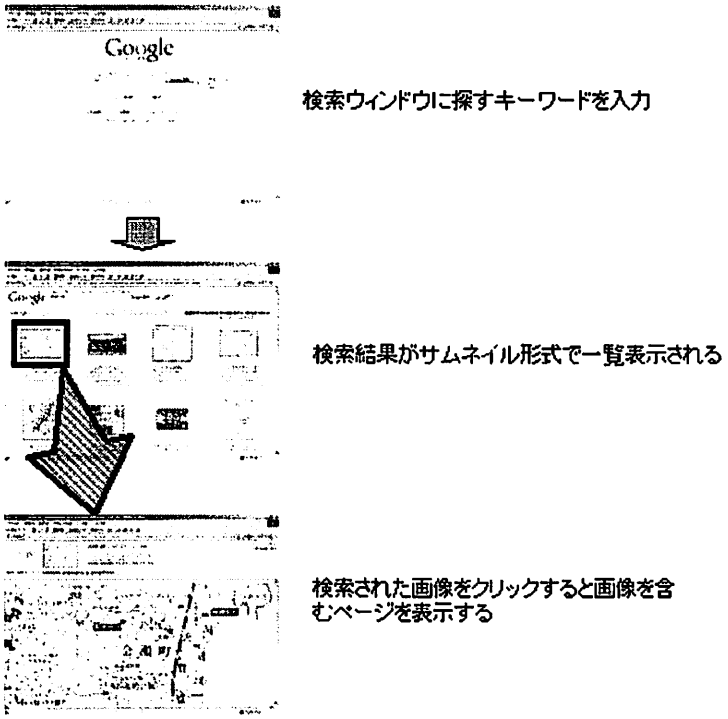


図 2.1: Googleイメージ検索の例

Google 画像検索の際のアルゴリズムは画像に隣接するテキスト、画像のキャプション、およびその他数多くの要因を分析して、コンテンツを判定し、高度なアルゴリズムを使用して重複を排除することによって、一番キーワードとのマッチングが高い画像が最初に表示される。しかし、その実体はテキスト分析による画像の索引化であり、画像そのものではなくHTML 文書中のメタ情報や、その他の文章といったものが検索されている。[1] Google 画像検索でも、Google のテキスト検索で使用するものと同じ演算子のすべてを使用でき、画像検索で「site:」を使用して特定の Web サイト上のイメージに検索を制限することも可能である。ここでメタ情報とは、HTML 文書の中で、その画像に与えられた意味情報であり、その文書の著者が明示的に与えた意味付け情報である。Google によって詳細な検索手法が公開されてはいないが、様々な実験の結果、img 要素の alt オプション内容のテキストがGoogle で索引化に使われていると考えられている。HTML 仕様上、Web ページ内で使用する画像には alt 文字列による代替文字列をつけることになっているが、実際のインターネット上では、そのような情報が一切なしに使われている画像も多い。そこでGoogle イメージ検索では、こうしたメタ情報を全く持たない画像についても、メタ情報以外の文章からも、間接的な情報として特定のキーワードと関連付け、検索の対象としている。また、検索キーワードと画像ファイルの距離が近い場合も、検索対象となっていると思われる。

2.2 テキストベースの検索手法

現在利用されているほとんどの画像検索はGoogle 画像検索のように検索ウインドウにキーワードを入力すると、キーワードに一致する画像データがサムネイル形式で一覧表示され、希望の画像をクリックすることにより、画像を含むページ全体が表示される。このようなキーワードによる画像検索はテキストベースによる画像検索でありクロスメディア検索の一つである。

一覧表示された各サムネイル画像の下には、その画像ファイルの名称、画像を含むサイトのURL、画像サイズなどの付加情報も表示される。

画像検索された画像は各検索サイト独自で保存している画像にインデックスを与えておき、保存されている画像の中から、検索キーワードを含むインデックスを持つ画像を、次のようなアルゴリズムに沿って検索している。各検索サイトに共通した主なアルゴリズムは、画像のキャプション、ページ内の画像の周辺テキスト、画像の alt 属性など画像の周辺情報内に検索ウィンドウに入力されたキーワードを含む場合にキーワードに関連する画像とし、検索された画像を各検索サイトの独自のアルゴリズムにより順位付けし表示している。

現在、インターネット上で公開されている多くの Web ページはテキスト主体のの構造となっており、画像の周辺には画像に強く関連するテキストが存在する。これらの画像周辺情報を利用することによって検索する画像検索手法をテキストベー

スのクロスメディア画像検索という。

2.3 現状の画像検索の問題点

現状の画像検索では、検索キーワードの要求と異なる画像が検索され、上位にランキングされる場合が存在する。検索キーワードと異なる画像が検索されてしまった例を図2.2に示す。

図 2.2: 誤検索の例

このような誤った検索結果が表示されてしまう原因として、検索キーワードが存在するだけで関連性のある画像と判断してしまうことが考えられる。しかし、実際には検索キーワードが出現していても全体的な文章は関連性が場合も多い。図2.2では野球選手である新庄剛志について「新庄剛志」で画像検索を行った結果、枠内の画像のように関連性の低い画像が検索されてしまっている。この原因は、新庄選手は野球選手であるので、野球に関連する内容の文書中に出現することが通常であるが、文章中に一度だけ出現してしまっている検索キーワードによって、全体的に他分野について書かれている文書であるにも関わらず検索されてしまった。この例のように、検索キーワードがWebページに含まれていたとしても、その画像を検索キーワードに関連する画像であるとは必ずしも言うことはできない。この点が現状の画像検索の問題点であると言える。

第3章　画像検索結果のフィルタリング

3.1　画像と文書内容の関連性

Web上には画像を含んだWebページが無数に存在する。Webページは作る製作者の持つなんらかの情報を発信するために作られたものが多い。そのような場合、そのWebページに含まれる画像はテキストと合わせ、情報伝達の手段の一つとして用いられる。そのため、使用されている画像は、Webページのテキストの内容と少なくとも何らかの関わりがある画像だと考えられる。つまり、Webページのテキスト内容と画像は密接な関係にあり、Webページに書かれている内容の分野は画像の分野と一致していると考えられる。このため、Webページの分野を推定できれば、画像の分野を推定できる。図3.1に本論文で画像とWebページの内容の関連性が高いと考える例を示す。

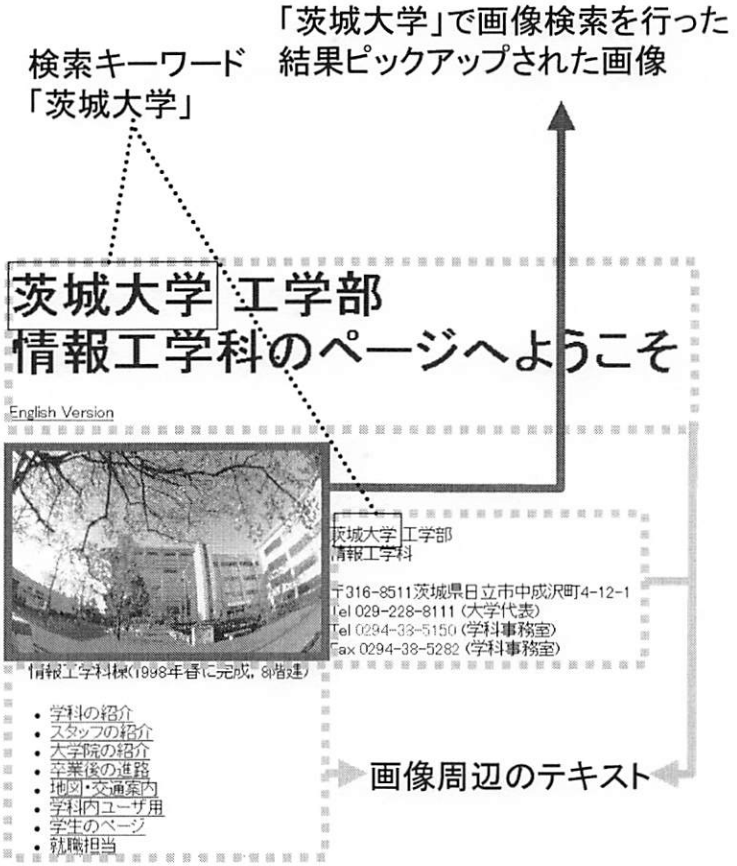


図 3.1: 画像と Web ページ内容が関連性が高いと考えられる例

この例は、「茨城大学」を検索キーワードに画像検索を行った結果、ピックアップされたある画像の画像元ページである。この Web ページを見ると中心にピックアップされた画像が存在し、また、検索キーワードである「茨城大学」も存在している事がわかる。周辺のテキスト中には「学科」、「大学院」、「情報工学科」「学生」など茨城大学という検索キーワードに関連が高い単語が出現している。このように、検索キーワードと関連性が高い単語が多く出現している Web ページの画像を、検索キーワードと画像の関連性が高い画像とした。

また、逆に関連性が低い Web ページの例を図 3.2 に示す。

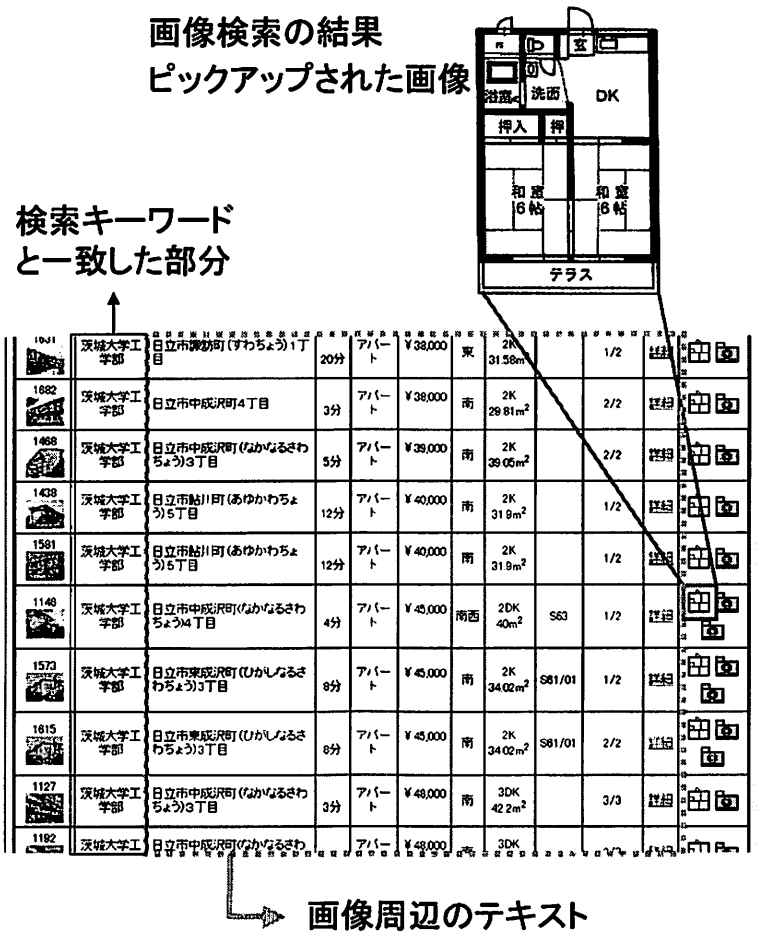


図 3.2: 画像と Web ページ内容が関連性が低いと考えられる例

この例は、前述の画像と同様に「茨城大学」を検索キーワードに画像検索を行った結果中の画像元ページである。この例では、確かに「茨城大学」という単語が存在しているが、画像周辺のテキスト中には先ほどの Web ページの様な関連性が高い単語はまったくと言っていいほど存在していない。このことから、この Web ページの画像は検索キーワードと関連性が低い画像であると言える。

3.2 文書分類を用いたフィルタリング

様々なカテゴリの文書が大量に混在して存在するとき、単語の出現頻度など、ある観点から文書のクラス分けを行う技術を文書分類という。文書分類を行った後

の文書の検索のイメージは、図書館の蔵書から意中の本を探すイメージと合致する。図書館の蔵書は棚によって歴史、社会、英語など分野ごとの棚に整理されている。この分野ごとに整理されていることによって、例えば戦国時代について述べられている本を読みたいと思った場合、歴史、日本史、と辿っていけば簡単に戦国時代に関する本を入手することができる。このように文書分類の基本的な利用法は、大量の文書を整理するために用いられる。

この文書分類は前述のように文書を整理する事が主な利用法だが、文書分類の手法を応用することによって特定の種の文書を排除するフィルタリングに利用することもできる。フィルタリングの例として、スパムメールの処理 [5] における文書分類の手法を説明する。まずスパムメールとは、受信者の意図を無視し、無差別かつ大量に一括して送信される、事前に許可していない広告などの受信者の必要としていない迷惑な電子メールのことを指す。このスパムメールが大量に送られてくることによって、サーバへの高い負荷がかかることによるサーバの不具合の発生や、受信者のメールを選別する作業時間の増加、添付ファイルを間違って開いてしまうことによるコンピュータウイルスへの感染など、様々な被害を被ることになる。このような被害を防ぐため、必要なメールとそうでない不必要なメールを自動的に選別、排除するフィルタリングが必要となる。このようなスパムメールのフィルタリングをするフィルタをスパムメールフィルタと呼ぶ。

このスパムメールフィルタリングには様々な手法やがあり、研究もされているが、一般的に使われる手法として文書分類の手法が用いられている。この文書分類を用いたスパムメールフィルタの作成は比較的容易に作成が出来、なお且つ良好なフィルタリング結果を得られることが知られている出現する単語によって判断する極単純なスパムメールフィルタについて説明する。まず、事前にスパムメールに使われる頻繁に、単語からスパムメールで出現する特徴的な単語を集めた判断基準となるデータを作成する。そして実際に送られてきたスパムメールの件名と本文から、繰り返し使用される言葉・特徴的な言葉を選び出し、事前に作成しておいた判断データと照らし合わせ、判断データ中の単語とマッチするという条件を満たすメールはスパムメールと判断し、排除もしくは特定のフォルダへの移動を行う。このようにして文書分類手法を利用したスパムメールのフィルタリングを行うことが出来る。文書分類を利用したスパムメールフィルタリングを説明図を図 3.3 に示す。

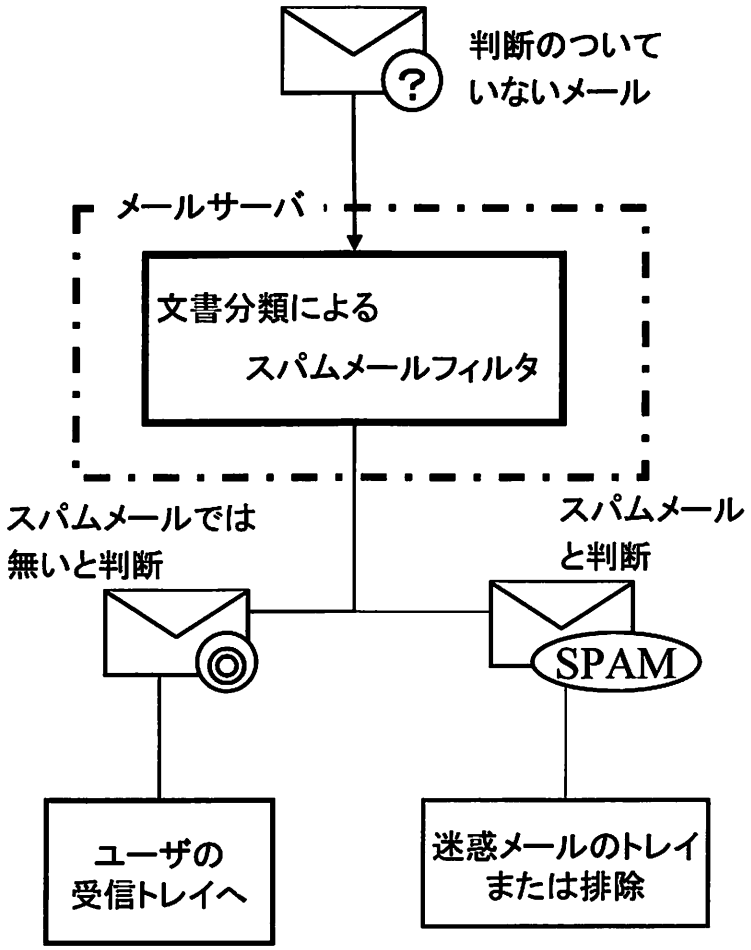


図 3.3: 文書分類によるスパムメールフィルタリング

3.3 Naive Bayes

3.3.1 確率の基本性質と条件付確率

まず、確率に関する基本的な性質について述べる。一般的に、実験や観測を試行と呼び、試行の結果として起こることがらが事象と呼ばれる。試行 T を行うとき、起こりうるすべての場合が n 通りあり、どの2つも同時に起こる事が無く、またどの場合の起こることも同程度に期待できるものとする。事象 A の起こるのは、この n 通りのうちのが a 通りであるとき、試行 T を行うときの、事象 A の起こる確率は次式のように定義される。

$$P(A) = \frac{a}{n} \tag{3.1}$$

この試行を行うときに起こる事象について、起こり得るすべてのことがらからなる事象を全事象と呼び、Ω で表す。また、事象 $A_1, A_2, A_3 \cdots$ が共に起こる事象を $A_1, A_2, A_3 \cdots$ の積事象といい、$A_1 \bigcap A_2 \bigcap A_3 \bigcap \cdots$ で表す。次に事象 $A_1, A_2, A_3 \cdots$ のうち少なくとも1つが起こる事象を $A_1, A_2, A_3 \cdots$ の和事象といい $A_1 \bigcup A_2 \bigcup A_3 \bigcup \cdots$ で表す。事象 A が起こらないことも1つの事象であり、これを A の余事象といい、$\bar{A}$ で表す。これらの事象の関係を集合の場合と同様にベン図で表したものを図3.4に示す。

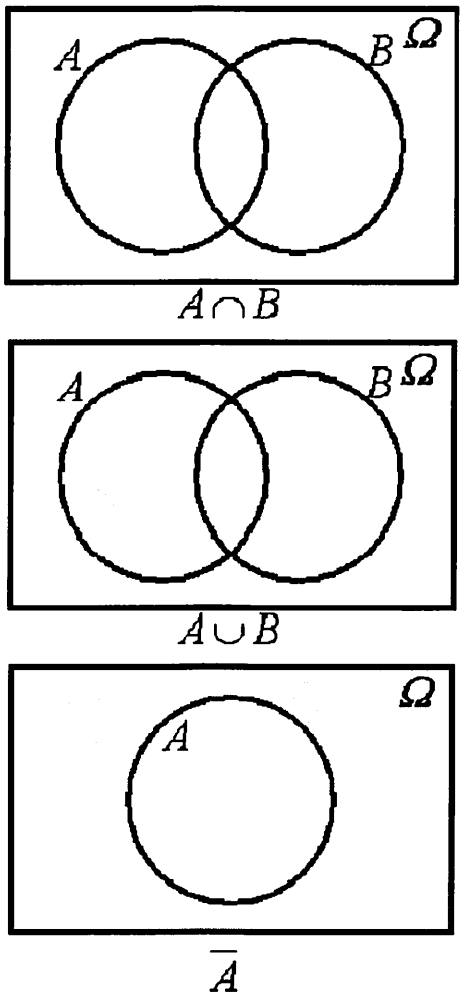


図 3.4: 事象の関係図

例えば、サイコロを振った場合1から6のいずれかの目が出ることから、この場合の全事象は次のようになる。

$$\Omega = \{1, 2, 3, 4, 5, 6\} \tag{3.2}$$

今、出る目の数が奇数の場合を事象Aとし、出る目の数が3以上の場合を事象Bとすると、$A \bigcap B$、$A \bigcup B$、$\bar{A}$は次のようになる。

$$A \bigcap B = \{3, 5\} \tag{3.3}$$

$$A \bigcup B = \{1, 3, 4, 5, 6\} \tag{3.4}$$

$$\bar{A} = \{2, 4, 6\} \tag{3.5}$$

次に出る目の数が4以上の偶数の場合を事象Cとすると$A \bigcap C$が起こることはありえない。このような起こることがありえないことも1つの事象とし空事象といい、ϕで表す。

また、事象$A_1, A_2, A_3 \cdots$があった場合、任意のi、$j(i \neq j)$について$A_i \bigcap A_j = \phi$であるとき、これらの事象は互いに背反であるといわれる。

次に条件付確率について述べる。トランプを例にして条件付確率について説明する。今よくきったトランプ52枚から1枚トランプを抜くとき、その札がスペードである事象をA、絵札である事象をBとすると起こりえるすべての場合は52通りであり次のようになる。

$$P(A) = \frac{13}{52}, P(B) = \frac{12}{52}, P(A \bigcap B) = \frac{3}{52} \tag{3.6}$$

このとき、抜いた札がスペードであることがわかっていると、起こりえるすべての場合がスペードの札の13通りに制限されるため、この中から抜いた札が絵札となる確率は$\frac{3}{13}$になる。

このように1つの試行を行うときに起こる事象A, Bに対して、$P(A) > 0$のとき、$P(A \bigcap B)/P(A)$を事象Aが起こったという条件のもとで事象Bの起こる条件付確率といい、$P(B \mid A)$と表す。式で表すと次のようになる。

$$P(B \mid A) = \frac{P(A \bigcap B)}{P(A)} \tag{3.7}$$

先ほどのトランプの例で、抜いた札がスペードであった条件のもとで、それが絵札である確率は式(3.7)から次のように算出される。

$$P(B \mid A) = \frac{\frac{3}{52}}{\frac{13}{52}} = \frac{3}{13} \tag{3.8}$$

また、式 (3.7) から $P(A) > 0, P(B) > 0$ のとき、次の確率の乗法定理が得られる

$$P(A \bigcap B) = P(A)P(B \mid A) = P(B)P(A \mid B) \tag{3.9}$$

3.3.2 ベイズの定理

ベイズの定理 [4] は今日では、幾つかの未観測要素を含むコンピュータによる推論等に応用され、迷惑メールの発見・分類といった作業の自動化（フィルタリング）といった情報工学上の情報ふるい分けに利用されている。この定理はある結果が得られた時、その結果に対する原因の確率を求める定理である。$P(A)$ を事象 A が発生する確率、$P(A|B)$ を事象 B が既に発生している場合に、事象 A が発生するという条件付確率とする。事象 B_i が発生した時、事象 A が発生する確率は $P(A|B_i)$ であるが、これを逆に見て、事象 A が発生したときに、B_i が発生していた確率 $P(B_i|A)$ を知りたい場合などに使われる。

今 $B_1, B_2, \cdots, B_n$ が互いに背反な事象で、どれかは必ず起こり、かつすべての i について $P(B_i) > 0$ とすると $P(A) > 0$ であるとする。図 3.5 に簡単な事象の関係図を示す。

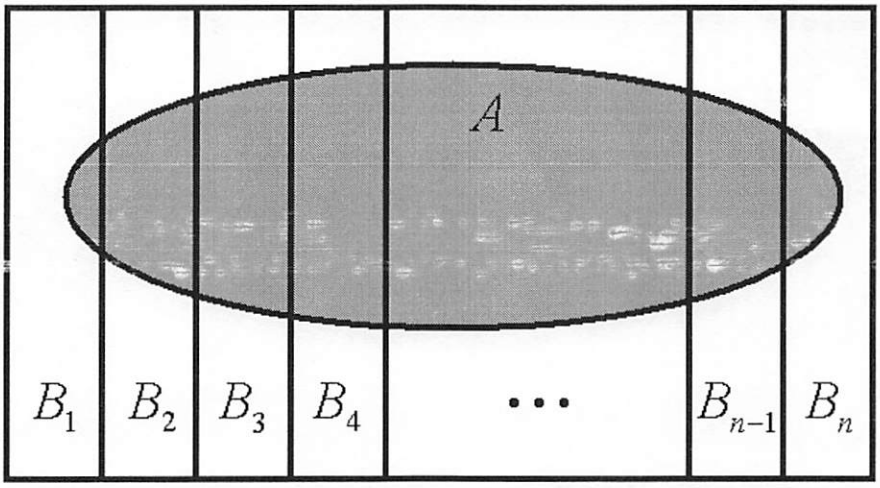


図 3.5: ベイズの定理の事象の関係図

事象 A は下記の様になる。

$$P(A) = \sum_{i=1}^{n} P(A \bigcap B_i) = \sum_{i=1}^{n} P(A \mid B_i)P(B_i) \tag{3.10}$$

式 (3.10) から確率 $P(B_i|A)$ は次のように表すことが出来る。

$$P(B_i \mid A) = \frac{A \bigcap B_i}{P(A)} = \frac{P(A \mid B_i)P(B_i)}{P(A)} = \frac{P(A \mid B_i)P(B_i)}{\sum_{i=1}^{n} P(A \mid B_i)P(B_i)} \tag{3.11}$$

この式 (3.11) をベイズの定理と呼ぶ。ベイズの定理において事象 B_i を原因系、事象 A を原因系により起こる結果とみる。このとき、各原因 B_i が結果 A を起こす確率 $P(A \mid B_i)P(B_i)$ が与えられると、逆に結果 A が起こった際に、それが原因 B_i により起こる確率 $P(B_i \mid A)$ というのがベイズの定理である。このような意味で $P(B_1), P(B_2), \cdots, P(B_n)$ を事前確率、$P(B_1 \mid A), P(B_2 \mid A), \cdots, P(B_n \mid A)$ を事後確率という。

3.3.3 Naive Bayes

ここでは、検索結果のフィルタリングに Naive Bays 法を用いた。Naive Bayes 法は全ての未知量を確率変数として扱い、未知パラメータも確率分布として推定を行う確率論に基礎をおく推定方法を用いた分類法であり、スパムメールの自動フィルタリングなどの文書分類において広く利用されている。その特徴としては、最終的な仮定成立の確率計算に事前知識を利用し、仮定が成立するかどうかを真か偽ではなく、確率で明示的に表現できる点にある。

ある事例 x を素性のリストとして、 $x = (f_1, f_2, \cdots, f_n,)$ とし、x の分類先のクラスの集合を $C = \{c_1, c_2, \cdots, c_m,\}$ としたとき、分類問題は $P(c \mid x)$ の分布を推定することで解決できる。実際に、 x のクラス c_x は以下の式で求まる。

$$c_x = \arg\max_{c \in C} P(c \mid x) \tag{3.12}$$

また、ベイズの定理より、

$$P(c \mid x) = \frac{P(c)P(x \mid c)}{P(x)} \tag{3.13}$$

となる。よって、式 (3.12)、式 (3.13) より、以下の式が成立する。

$$c_x = \arg\max_{c \in C} P(c)P(x \mid c) \tag{3.14}$$

しかし、$P(c)$ は比較的容易に推定できるが、$P(x \mid c)$ の推定は現実的に難しいため、Naive Bayes のモデルでは以下の仮定を導入する。

$$P(x \mid c) = \prod_{i=1}^{n} P(f_i \mid c) \tag{3.15}$$

$P(f_i \mid c)$ の推定は比較的容易であるために、結果として $P(x \mid c)$ が推定できる。Naive Bayes を使った分類がうまくいくためには式 (3.15) の仮定をできるだけ満たすような素性を選択することが必要である。

この Naive Bayes 法を利用して文書分類を行う分類器を Naive Bayes 分類器という。Naive Bayes 分類器の概要を表した図を図 3.7 に示す。

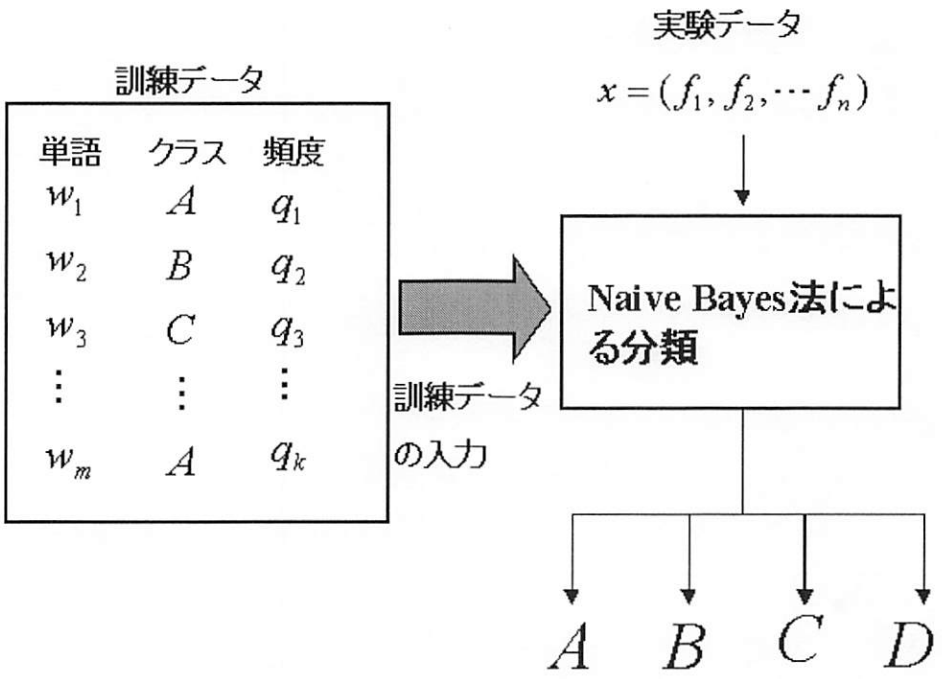


図 3.6: Naive Bayes 分類器

3.4 画像を含む HTML 文書の構造

現在一般的に公開されている Web ページのほとんどが HTML によって記述されている。HTML は HyperText Markup Language の略称で、その特徴はタグと呼ばれる要素を用いて Web ページの構造を作成することが挙げられる。

HTML には様々なタグが存在するが、HTML で画像を挿入するためには IMG タグというタグが用いられる。IMG タグは次のように記述される。

$< IMG \quad SRC =$ "表示させたい $JPEG$ や GIF などの画像ファイル名"

$ALT =$ "代替えテキスト" $WIDTH =$ "画像の幅 $HEIGHT =$ "画像の高さ" $>$

IMG タグの内側にある SRC、ALT などは要素の属性と呼ばれる部分である。SRC 属性には表示したい画像のあるアドレスを記述する。ALT 属性とは代替えテキストと呼ばれる、画像が表示不可能であった場合などに表示する文章を記述する。この文章は画像の上にマウスポインタもって行くことによって表示することができる。直接的に画像について記述している文章であるといえる。

画像を含む Web ページの HTML 文書は、主に図 3.7 のような構造となっている。

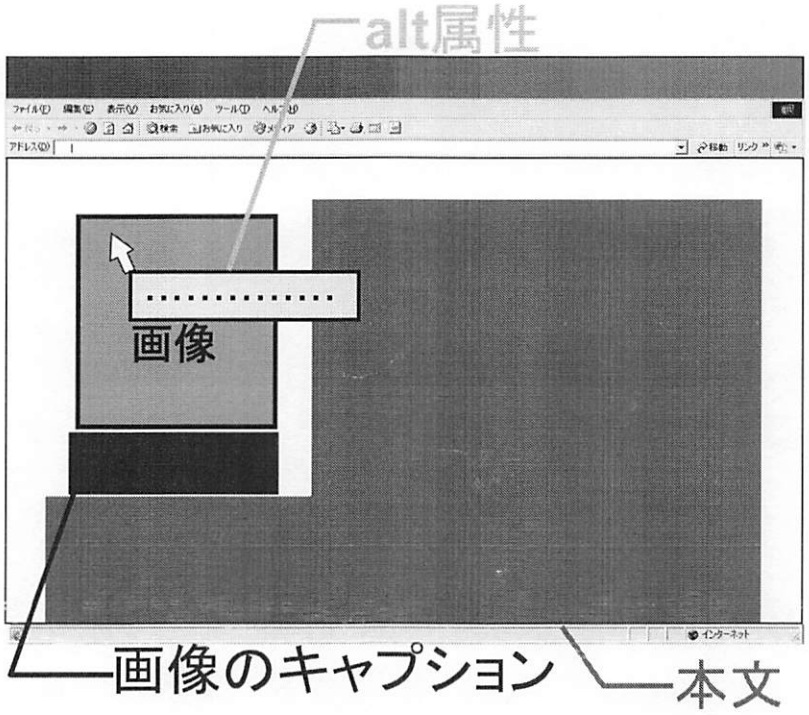


図 3.7: 画像を含む HTML 文書の構造の例

第4章　実験

4.1　使用データ

今回は対象となる画像の種類を限定し、画像の中でも特に需要が高いと思われる、人名による画像検索に特化して実験を行った。実験に使用した人名は「イチロー」、「小泉純一郎」、「渡辺謙」を使用した。これらの人物名を実験データの対象とした理由として、スポーツ、政治、芸能の各分野で昨年大いに活躍し、様々なメディアに多く登場した各分野を代表する人物であるということで実験の対象人名としている。

また、今回Naive Bayes分類器を使用するに当たって、使用したデータは大きく分けて2種類のデータを用いた。まず、Naive Bayes分類器によってクラス分けするために用いる訓練データと、実際に入力する実験データである。

トレーニングデータは、人名をテキスト検索によって得たHTML文書群を使用した。実験データにはGoogleイメージ検索によって対象人名を検索した結果を利用した。

このように検索によって得られたHTML文書はトレーニングデータ、実験データ共に次のような処理を行う。まずHTML文書からHTMLのタグを外し、本文のみにする。次に形態素解析を行い出現する品詞ごとに分ける。形態素解析とは、コンピューター等を用いた自然言語処理の基礎技術のひとつで、自然言語で書かれた文を形態素という言語で意味を持つ最小単位の列に分割し、品詞を見分けることである。この形態素解析には今回京都大学のJUMANを使用した。次に形態素解析を行った結果から、名詞のみを抽出した。そして、トレーニングデータでは検索によって得たHTML文書群から作成し、実験データでは各検索された画像元のHTML文書について重複を除き出現頻度順にまとめたデータを作成した。このデータ作成処理の流れを示した例を図4.1に示す。

図 4.1: データ作成処理の流れの例

実際に実験データを入力する前に、まず Naive Bayes による学習を行う。学習される分類器の入力は前述の処理で作成されたトレーニングデータの入力を行う。識別するため作成するクラスは「イチロー」「小泉純一郎」「渡辺謙」「その他」の4つである。訓練データは「イチロー」「小泉純一郎」「渡辺謙」をキーに Google で検索を行い、その検索結果の HTML 文書群を用いそれらのクラスの訓練データとした。「その他」のクラスの訓練データとしては国際、経済、社会の HTML 文書を適当に用意した。実験データは各実験項目にて異なるデータを用いている。これは Google のイメージ検索結果が更新されているためである。

4.2 重み付け無しによる実験

まず実験データとして入力するデータにテキストの個所による重み付けをせずに実験を行った。この実験で使用した訓練データは、「イチロー」、「渡辺謙」、「小泉純一郎」についてキーワード検索を行い、その結果得られた上位各 40 件の HTML 文書からそれぞれの人名に関する訓練データを作成した。また、社会、経済、国際の各 20 件の HTML 文書からその他に属する訓練データを作成した。データ作成には 4.1 の処理を施し作成した。各クラスのトレーニングデータの出現頻度の高い部分は次ページの表 4.1、表 4.2、表 4.2、表 4.4 のようになっている。

表 4.1: イチローのトレーニングデータの出現頻度上位

1765	イチロー	145	一	75	■	50	だ
966	安打	140	戦	73	日記	50	5
737	記録	139	,	73	成績	50	262
468	日	138	ー	71	258	49	週刊
340	試合	131	シスラー	71	URL	49	レンジャーズ
337	-	130	TB	70	っ	49	トラバ
325	年	127	ヒット	69	投稿	49	C
324	達成	124	—	68	私	48	賞
321	新記録	122	キーワード	68	プロ	48	October
302	選手	117	リーグ	67	中	47	祝
294	1	116	from	67	♪	46	日程
280	打席	109	★	66	―	46	スポーツナビ
262	at	105	松井	63	投手	45	打率
260	–	104	スポーツ	63	ファン	45	何
249	2	103	本	63	コメント	45	257
238	最多	103	人	62	””	44	鈴木
233	3	100	シアトル	61	時分	44	Yahoo!
229	更新	96	連続	61	概要	44	ICHIRO
227	こと	96	米	61	チーム	43	頭
223	野球	94	ブログ	60	ブログタイトル	43	世界
212	シーズン	93	ニュース	60	...	43	プレッシャー
209	.	91	本人	59	時間	43	ジョージ
205	年月日	89	打者	57	情報	42	瞬間
204	マリナーズ	88	259	57	ゲスト	42	]
197	バック	87	4	57	もの	41	打点
190	大リーグ	86	..	57	30	41	結果
186	メジャー	85	歴史	57	…	41	84
183	トラック	81	打	56	通算	41	^
180	記事	80	打数	55	テレビ	40	優勝
175	今日	79	・	54	彼	40	毎日
172	年間	78	土	53	オリックス	40	a
170	月	78	タイトル	52	首位	40	[
159	、	77	外野手	52	7	39	監督
154	の	77	’	51	樹立	39	感動
150	日本	76	年月	50	出場	…	…

表 4.2: 小泉純一郎のトレーニングデータの出現頻度上位

807	小泉	82	年	50	ウッソ	36	森
397	純	80	国	48	—	36	三
385	一郎	77	週刊	47	米	36	解散
383	こと	77	事	46	本人	36	ぼ
252	日本	71	アメリカ	46	選挙	35	保障
240	鬱	70	ハム	46	真紀子	35	等
220	,	70	-	46	国家	35	誰
196	[]	69	政権	45	著者	35	春秋
194	日	69	な	45	PM	34	藝
190	人	69	@	44	3	34	厚生
185	政治	68	常勝	43	本	34	よ
185	-	68	軍団	43	方	34	●
180	首相	66	北朝鮮	42	政府	34	sYVUuuprvTv
174	の	66	カテジナ	42	自衛隊	33	僕
169	内閣	65	物書き	41	発言	33	中
157	大臣	65	推敲	41	年金	33	地域
145	総理	64	ー	41	国債	33	新聞
139	—	63	民営	41	外交	33	情報
138	私	63	だ	40	社会	33	場合
132	一	62	議員	40	自分	33	金
129	改革	60	December	40	時代	32	男
126	キーワード	59	支持	40	参拝	32	事件
109	国民	58	顔	40	お	32	事業
105	経済	58	、	40	2	32	最近
104	.	54	戦争	39	話	32	公約
101	もの	54	何	39	発行	32	気
99	イラク	54	■	38	制裁	31	出版社
96	年月	54	,ISBN	38	衆議院	31	構造
95	位	52	田中	38	会	31	り
93	…	52	総裁	37	人間	30	拉致
89	郵政	52	A	37	号	30	法
85	っ	51	文	37	ん	30	関連
84	年月日	50	靖国	37	ところ	30	意見
84	自民党	50	問題	37	うん	30	あなた
84	ため	50	批判	36	世界	…	…

表 4.3: 渡辺謙のトレーニングデータの出現頻度上位

284	渡辺	50	ページ	30	ちてん	23	政
284	謙	49	by	29	感	23	賞
277	マン	48	件	29	ㄴ	23	関連
242	バット	47	日本	29	.	23	ヒーロー
190	映画	47	シリーズ	28	笑	23	テレビ
155	,	46	通常	28	もの	22	竜
152	ビギンズ	46	バッドマン	28	っ	22	梅
150	年月日	43	発送	28	¥	22	事
136	・	42	Yahoo!	28	[]	22	一覧
124	税	41	ドラマ	27	年月	22	以内
120	時分	40	年	27	新品	22	クルーズ
120	-	40	公開	27	主演	22	..
119	日	40	ハリウッド	27	今	21	物語
102	秒	40	クリスチャン	27	感想	21	日記
100	¥,	38	サイト	27	リーアム	21	藤枝
100	DVD	38	–	27	スポーツ	21	場合
95	の	35	役	26	続き	21	作
81	検索	35	ブルース	26	更新	21	眼
78	...	34	情報	26	限定	21	会
75	人	34	ぷ	26	月日	21	闇
74	監督	33	サムライ	26	円	21	モーガン
73	込	33	、	25	商品	21	マイケル
72	こと	33	at	25	気	21	Pt
66	価格	32	北	25	ランキング	20	話題
61	版	32	米	25	■	20	独
61	作品	32	一	25	’	20	時
59	発売	32	♪	25	% OFF!	20	仕掛
59	出演	32	‥	24	宗	20	公式
58	]	31	零	24	今回	20	系列
57	[	31	ラン	24	安	20	感じ
55	—	31	クリストファー	24	マーケット	20	ユーズド
53	俳優	31	……	24	プレ	20	タイトル
53	私	30	時間	24	ニーソン	20	ケイン
51	結果	30	ラスト	24	トム	20	ー
50	抜	30	ベール	24	イス	…	…

表 4.4: その他のトレーニングデータの出現頻度上位

520	年月日	40	海外	22	平均	20	]
180	-	40	ヘルプ	22	分	20	[
120	水	40	トピックス	22	IP	20	Yahoo
120	時分	40	エンターテインメント	21	比	20	Rights
111	経済	40	Copyright	21	提携	20	Reserved.
100	記事	40	C	20	問い合わせ	20	PC
100	火	39	ロイター	20	無断	20	Japan
100	ニュース	38	産	20	排除	20	JAPAN
81	年月	36	新聞	20	年代	20	HD
81	日	36	経	20	転用	20	Corporation.
80	木	35	産業	20	創業	20	All
80	日月	35	株式	20	前後	20	–
80	土	35	悪化	20	条件	19	判断
80	金	34	修正	20	市況	19	東京
74	事業	34	月	20	高知	19	市場
72	買収	32	街角	20	更新	19	回復
69	パソコン	32	Yahoo!	20	固定	19	ＩＢＭ
66	総合	31	再生	20	原案	18	9
61	写真	30	速報	20	掲示板	18	3
60	検索	30	ダイエー	20	競馬	17	府
60	トップ	29	店	20	去年	17	発売
60	IBM	29	8	20	過	17	内閣
55	景気	28	連続	20	影響	17	大子
52	聯	28	閉鎖	20	一覧	17	週刊誌
52	想	27	大手	20	意見	17	皇
51	通信	26	電話	20	位	17	延期
49	米	26	値	20	レノボグループ	15	発表
48	中国	26	銭	20	ライブ	15	日銀
48	コンピュータ	25	世界	20	ミサワ	15	調整
43	地域	24	実質	20	ドア	15	前年
43	国内	24	共同	20	シェア	15	生産
42	スポーツ	24	あなた	20	カスタマイズ	15	期
41	時事	24	ＧＤＰ	20	7	15	こと
40	動画	23	企業	20	11	14	日経
40	月日	23	下方	20	・	…	…

次に Google で「イチロー」、「小泉純一郎」、「渡辺謙」をそれぞれキーに画像検索した結果の各上位 20 件、中位 20 件、下位 20 件の画像と HTML 文書を取り出し、画像のフィルタリング実験を行った。データ作成には 4.1 の処理を施し作成した。その HTML 文書を先の学習できた分類器により、その文書が「イチロー」「小泉純一郎」「渡辺謙」「その他」のどのクラスに属するかのスコアを求めた。その合計スコアーの「イチロー」に対するスコアーの比率をその文書の「イチロー」に対する関連度とした。合計スコアーを Sum 、そのテキストのクラスのスコアー $Score$ から、次式により各テキストファイルごとの各クラスに属する比率 $Ratio$ を求める。算出式は次式のようになる。

$$Ratio = \frac{Score}{Sum} \tag{4.1}$$

この関連度により、Google の検索結果のランキングを変更した。「小泉純一郎」「渡辺謙」に対しても同様の実験を行った。この実験の流れを図 4.2 に示す。

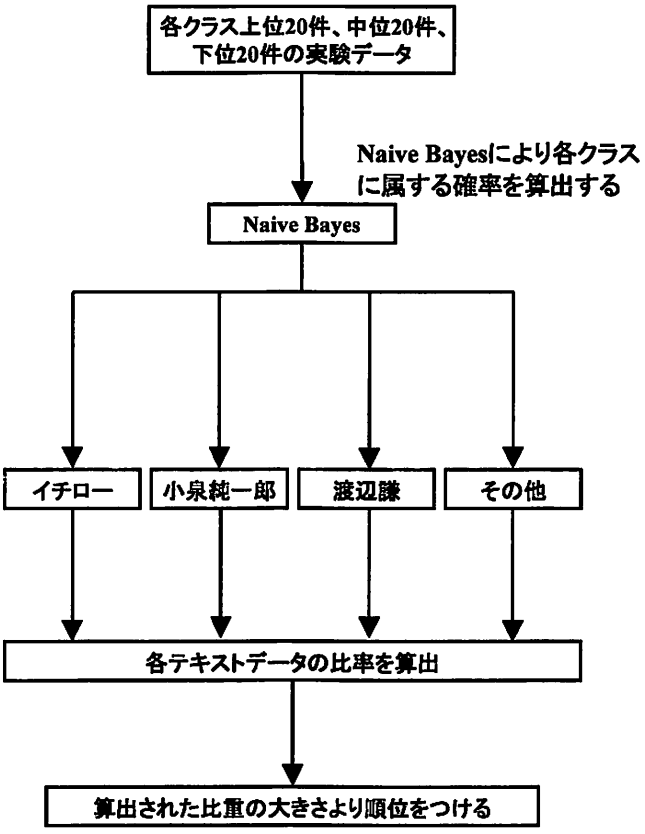


図 4.2: 重み付け無しの実験のフローチャート

この実験を行う前のGoogleによるイチローの画像検索結果の上位中位下位の順位を, 図4.3、図4.4、図4.5にそれぞれ示す。小泉純一郎、渡辺謙の検索結果は付録に添付する。

図 4.3: イチローの画像検索結果上位

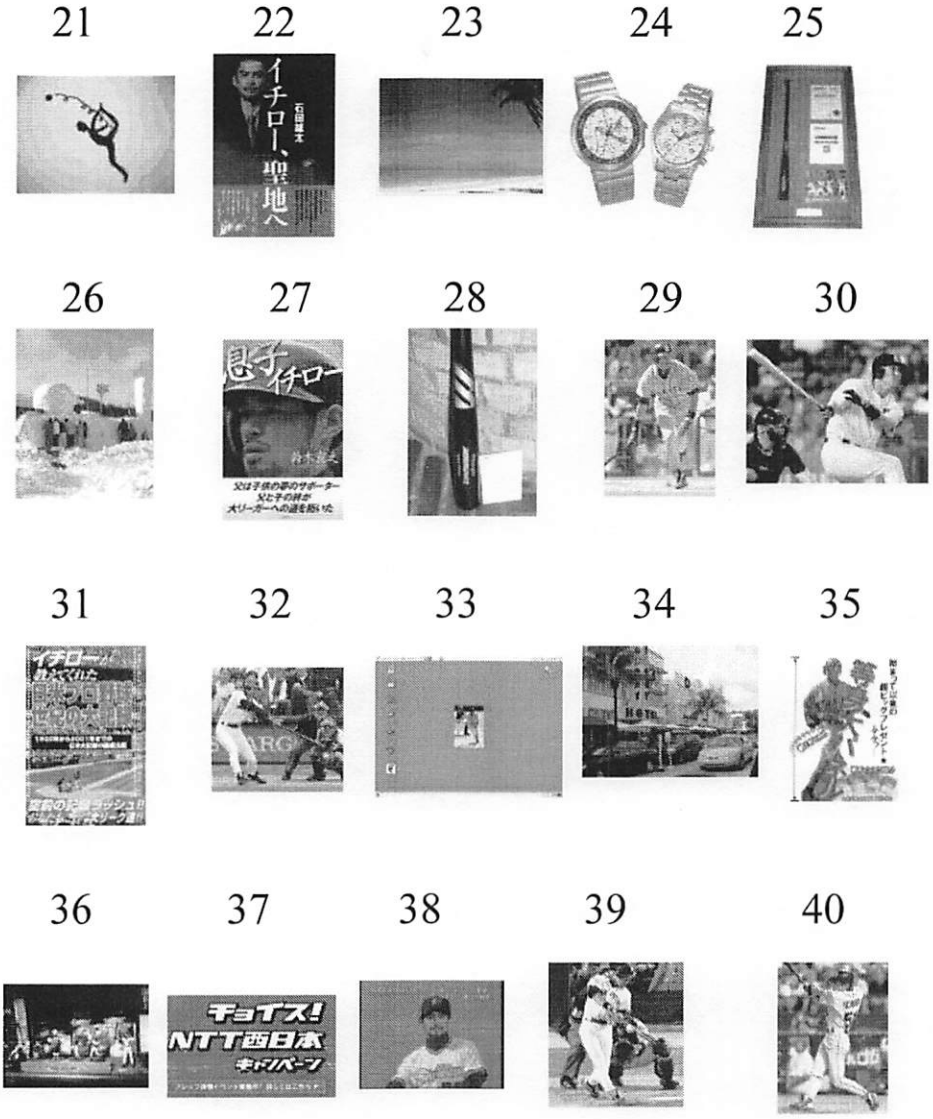


図 4.4: イチローの画像検索結果中位

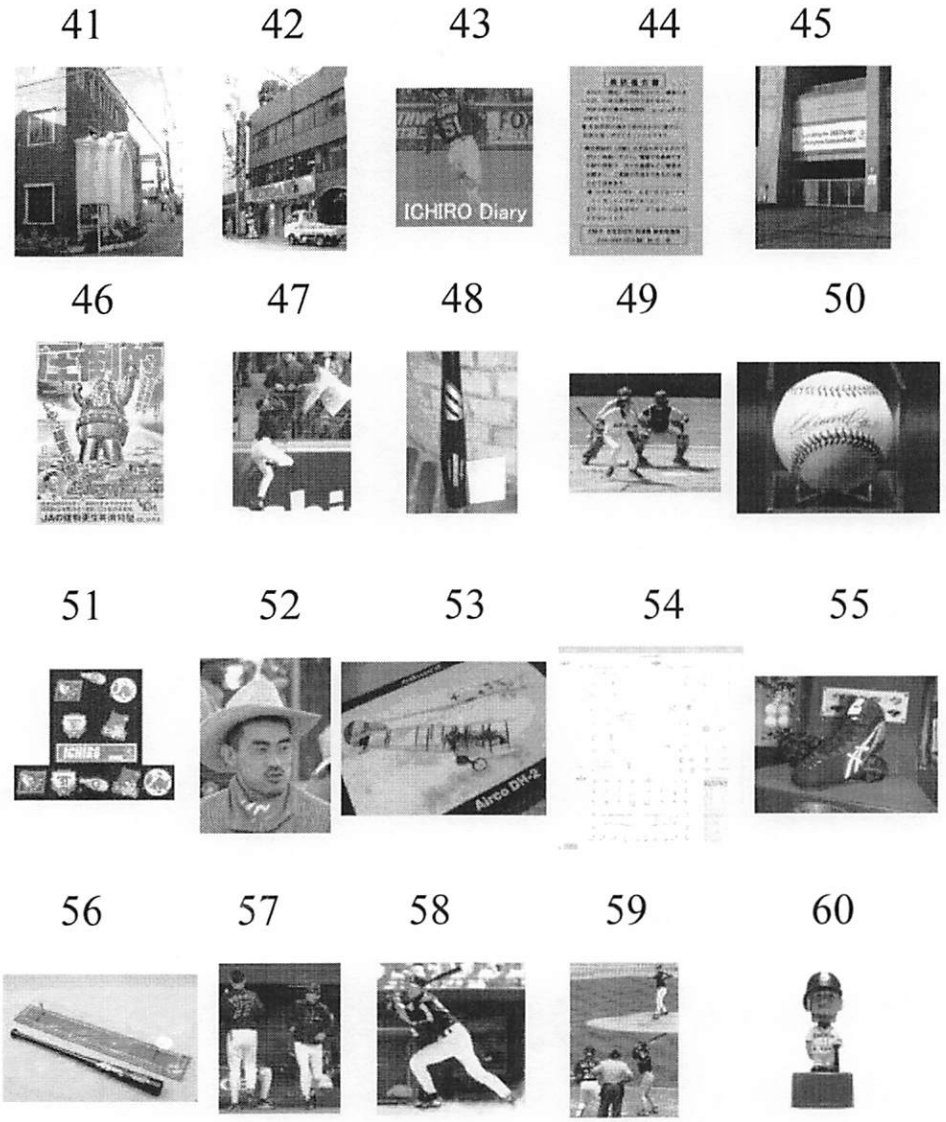


図 4.5: イチローの画像検索結果下位

これらのイメージ検索結果に対し本実験を施した後の結果を次に示す。ここではイチローの実験結果のみ示す。小泉純一郎、渡辺謙の結果については付録に添付する。

図 4.6: イチローの重み付け無しフィルタリング実験結果 1

図 4.7: イチローの重み付け無しフィルタリング実験結果 2

図 4.8: イチローの重み付け無しフィルタリング実験結果 3

4.3 ALT 属性に対し重みをつけて行った実験

次に HTML 文書の ALT 属性に注目し、ALT 属性に対し重み付けをして実験を行った。ALT 属性は画像に意味を持たせるもので、「代替テキスト」とも呼ばれ、画像挿入のため img 要素を使用した場合には ALT 属性を必ず記述する必要がある。記述する内容は img 要素の画像の内容について記述しなければならないが、「alt="" 」と表記すると ALT 属性は空白になる。本来はブラウザに画像が表示されない場合に、画像の替わりに文字でそこに何があったのかを説明するものであるが、画像が表示されている場合は、その画像にマウスポインタをのせると ALT 属性の内容が表示される。このように、ALT 属性は HTML 文書の中でも画像について直接の説明がなされていることが多い部分である。また、ニュースサイトなどの信用性の高い Web ページであるほど画像について簡潔で直接的な説明してい

る特徴が見られる。

そこで、このALT属性の内容に着目し、この部分に出現する単語群に対し重み付けをした実験を行った。実際の重み付けの方法はALT属性に出現した単語群をNaive Bayes分類器に入力した結果から求めたスコアと、Webページの本文に出現した単語群からも同様にスコアを求め、ALT属性のスコアに重み付けをし、本文のスコアに加算したスコアの大小によってランキング付けを行った。なお、この実験の訓練データ、及びスコアの算出方法は4.2と同様である。

この実験の訓練データでは、Naive Bayes分類器に入力する訓練データを訓練データの正確性を高めるため、Webページの内容を吟味して厳選したものにした。表4.5、表4.6、表4.7、表4.8に各クラスの訓練データを示す。

表 4.5: イチローの訓練データ

112	イチロー	15	チーム	8	名	6	四球
97	安打	15	スポーツ	8	本	6	球団
95	.	15	の	8	打点	6	球
75	年	14	打数	8	時代	6	英語
57	選手	14	史上	8	現地	6	ファン
44	マリナーズ	14	シアトル	8	リンク	6	アメリカ
41	こと	14	5	8	タイトル	6	ところ
36	–	13	打率	8	コーチ	6	8
33	記録	13	最多	8	もの	6	—
32	リーグ	13	王	8	OAK	5	野村
32	-	12	編集	7	塁	5	捕手
29	メジャー	12	年月日	7	木	5	当時
28	打者	12	賞	7	評価	5	電
27	試合	12	監督	7	日記	5	地元
27	オリックス	12	移籍	7	内野	5	打法
27	1	12	]	7	得点	5	人目
26	日本	12	[	7	成績	5	新人王
23	達成	11	戦	7	出演	5	情報
22	シーズン	10	度	7	三振	5	松井
21	打席	10	初	7	現在	5	勝
20	連続	10	時間	7	月	5	首脳
20	3	10	更新	7	ホームラン	5	守備
19	野球	10	一	7	ページ	5	紙
19	日	10	┗	7	プレー	5	仰
19	首位	10	プロ	7	ヒット	5	軌跡
18	4	10	シスラー	7	ジョージ	5	割
18	2	9	米	7	ア	5	位
16	盗塁	9	年間	7	ため	5	ベースボール
16	投手	9	打撃	7	TEX	5	ヘルナンデス
16	通算	9	出場	7	SEA	5	ピッチャー
15	本塁打	9	月間	6	番	5	スター
15	本人	9	結果	6	年度	5	ゴールドグラブ
15	打	9	軍	6	入団	5	つ
15	今季	9	外野手	6	新記録	5	TODAY
15	獲得	9	★	6	受賞	…	…

表 4.6: 小泉純一郎の訓練データ

96	小泉	8	次	5	大学	4	聖域
57	純	8	行	5	選挙	4	人質
52	-	8	厚生	5	選	4	集
51	一郎	8	関連	5	政権	4	首脳
36	内閣	8	改造	5	初	4	事
31	年月日	8	解散	5	春秋	4	参議院
31	–	7	代	5	実現	4	三
26	大臣	7	自由民主党	5	市	4	作成
24	年	7	支持	5	国家	4	際
19	年月	7	号	5	国	4	国債
19	総理	7	会議	5	構造	4	検索
18	編集	6	文	5	曲	4	経済
17	]	6	父	5	帰国	4	記念
17	[	6	人	5	会談	4	官邸
16	首相	6	神奈川	5	会社	4	解放
16	自民党	6	自衛隊	5	加入	4	バラード
16	こと	6	参拝	5	岡田	4	テロ
15	改革	6	国民	5	ページ	4	ツール
15	ISBN	6	国会	5	ため	4	な
14	郵政	6	更新	5	—	4	Love
14	衆議院	6	回	5	X	4	JAPAN
14	議員	6	横須賀	4	拉致	4	Forever
13	日本	6	一	4	落選	4	skins-.monobookIEFixes.css@import
13	政治	6	ファン	4	法	3	龍太郎
12	民営	5	藝	4	米	3	問題
12	年金	5	論	4	批判	3	民主党
12	総裁	5	靖国	4	版	3	万
12	—	5	也	4	発言	3	保険料
11	リンク	5	本部	4	当選	3	変人
10	就任	5	本人	4	大会	3	米国
10	公約	5	北朝鮮	4	祖父	3	福田
10	イラク	5	法案	4	戦略	3	評価
10	・	5	表示	4	戦闘	3	表明
9	発足	5	発表	4	戦後	3	百
9	日	5	地域	4	先	…	…

表 4.7: 渡辺謙の訓練データ

44	渡辺	7	マン	4	円	3	証明
38	謙	7	シリーズ	4	ラストサムライ	3	松竹
36	年月日	6	物語	4	ラスト	3	少年
36	年	6	人	4	タンポポ	3	出身
30	–	6	仕掛	4	サムライ	3	時
23	-	6	公開	4	わた	3	殺
22	・	6	関連	4	けん	3	更新
20	系列	6	ビギンズ	4	TBS	3	元
18	宗	6	バット	4	SAYURI	3	掲載
18	監督	5	役	4	C	3	魚
17	政	5	日記	3	絆	3	球団
15	竜	5	藤原	3	話題	3	記憶
15	東映	5	町	3	恋	3	厳
15	Yahoo!	5	壬生	3	零	3	下谷
14	独	5	新潟	3	流	3	一郎
14	眼	5	士	3	野	3	一
14	映画	5	阪神	3	明日	3	杏
12	版	5	演劇	3	万	3	ヘラルド
12	フジテレビ	5	ヘルプ	3	幕末	3	プロフィール
12	NHK	5	ハリウッド	3	北	3	ファン
11	ドラマ	5	なべ	3	謀	3	デビュー
10	月日	4	白血病	3	編	3	タレント
9	日本	4	年月	3	武蔵	3	ケイダッシュ
9	大河	4	東宝	3	表示	3	な
9	テレビ	4	天	3	発売	3	kg
9	]	4	男性	3	島	3	cm
9	[	4	千	3	蔵	3	Wikipedia
8	編集	4	小次郎	3	戦い	3	Rights
8	検索	4	出演	3	生	3	Reserved.
8	リンク	4	集団	3	瀬戸内	3	JAPAN
7	梅	4	集	3	仁義	3	Copyright
7	俳優	4	写真	3	人名	3	All
7	藤枝	4	作品	3	人間	3	A
7	伝	4	義	3	身長	2	蜷川
7	安	4	海	3	情報	…	…

表 4.8: その他の訓練データ

250	-	10	ヘルプ	6	情報	4	等
164	日	10	トップ	6	最大	4	投資
159	時分	10	スポーツ	6	検索	4	転職
40	年月日	10	サイエンス	6	銀	4	知識
35	経済	10	コンピュータ	6	ＦＲＢ	4	大手
33	新聞	10	エンターテインメント	5	預金	4	政権
27	ロイター	10	ＮＹ	5	問い合わせ	4	世界
25	通信	9	貿易	5	木	4	人
24	米	9	日報	5	分	4	新報
24	Yahoo!	9	東	5	動画	4	証券
23	銀行	9	中国	5	長官	4	債権
23	円	9	相場	5	月日	4	高値
19	金融	9	国際	5	急騰	4	午前
17	毎日新聞	9	海外	5	記事	4	計画
17	共同	9	円相場	5	介入	4	議長
17	為替	8	米国	5	過去	4	基礎
17	1	8	時	5	ブッシュ	4	改革
16	外為	8	財務	5	カスタマイズ	4	解説
15	写真	8	企業	5	43	4	ドル
14	読売	8	外国	5	33	4	インフレ
14	東京	7	日本	5	3	3	連続
14	銭	7	政府	5	Yahoo	3	連合
14	市場	7	時事	5	Rights	3	抑制
14	ニュース	7	決算	5	Reserved.	3	目標
13	神戸	7	金利	5	Japan	3	統合
12	117	7	機関	5	JAPAN	3	冬
11	地域	7	欧州	5	Corporation.	3	電話
11	政策	7	奥	5	Copyright	3	田中
11	国内	7	一	5	C	3	注目
11	更新	7	アメリカ	5	BB	3	中間
11	株式	7	116	5	.	3	続
11	トピックス	7	・	4	利上げ	3	制度
10	赤字	7	All	4	利益	3	人民
10	円高	6	日本銀行	4	問題	3	伸
10	ユーロ	6	通貨	4	動き	…	…

この実験の実験データは、各検索キーワードによるGoogleイメージ検索結果から、各20件のWebページを抽出した物を実験データとした。作成の実験データを図4.9、図4.10、図4.11にそれぞれ示す。

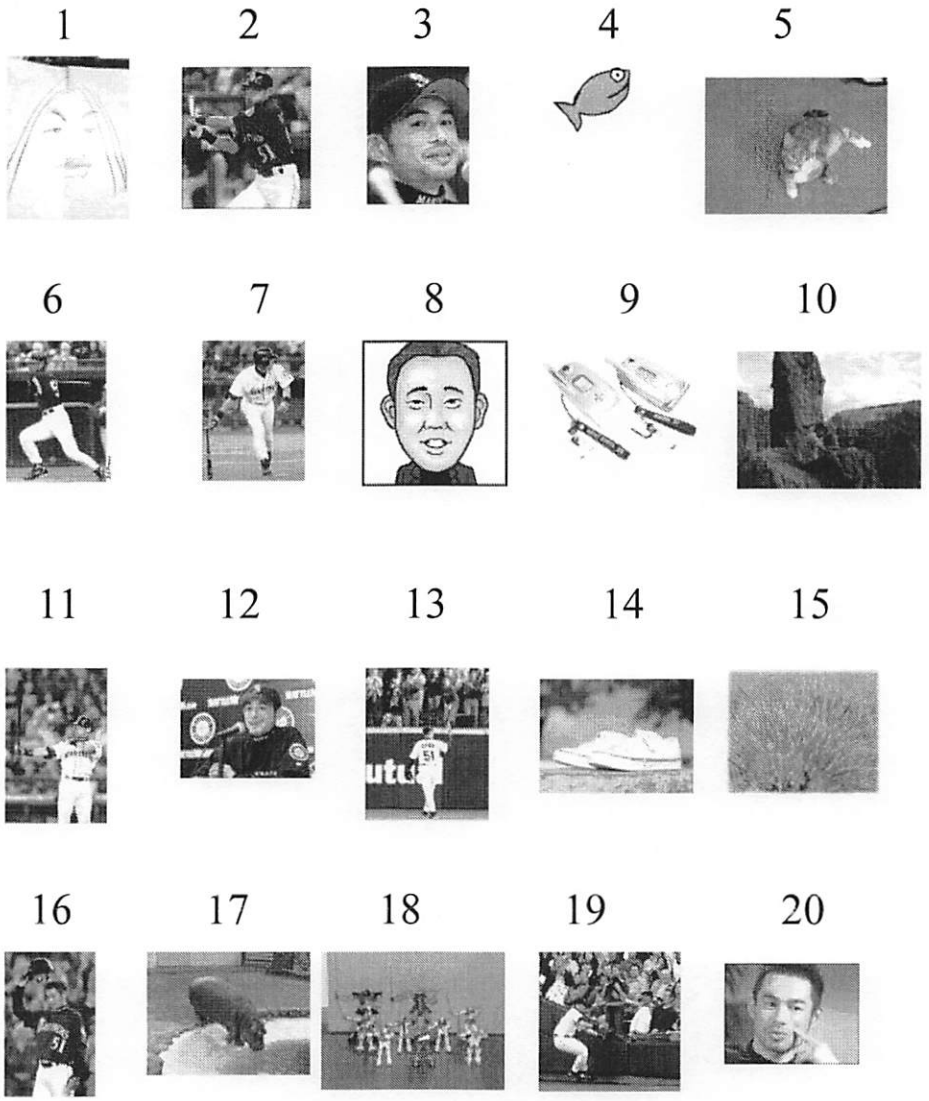


図 4.9: イチローの実験データ

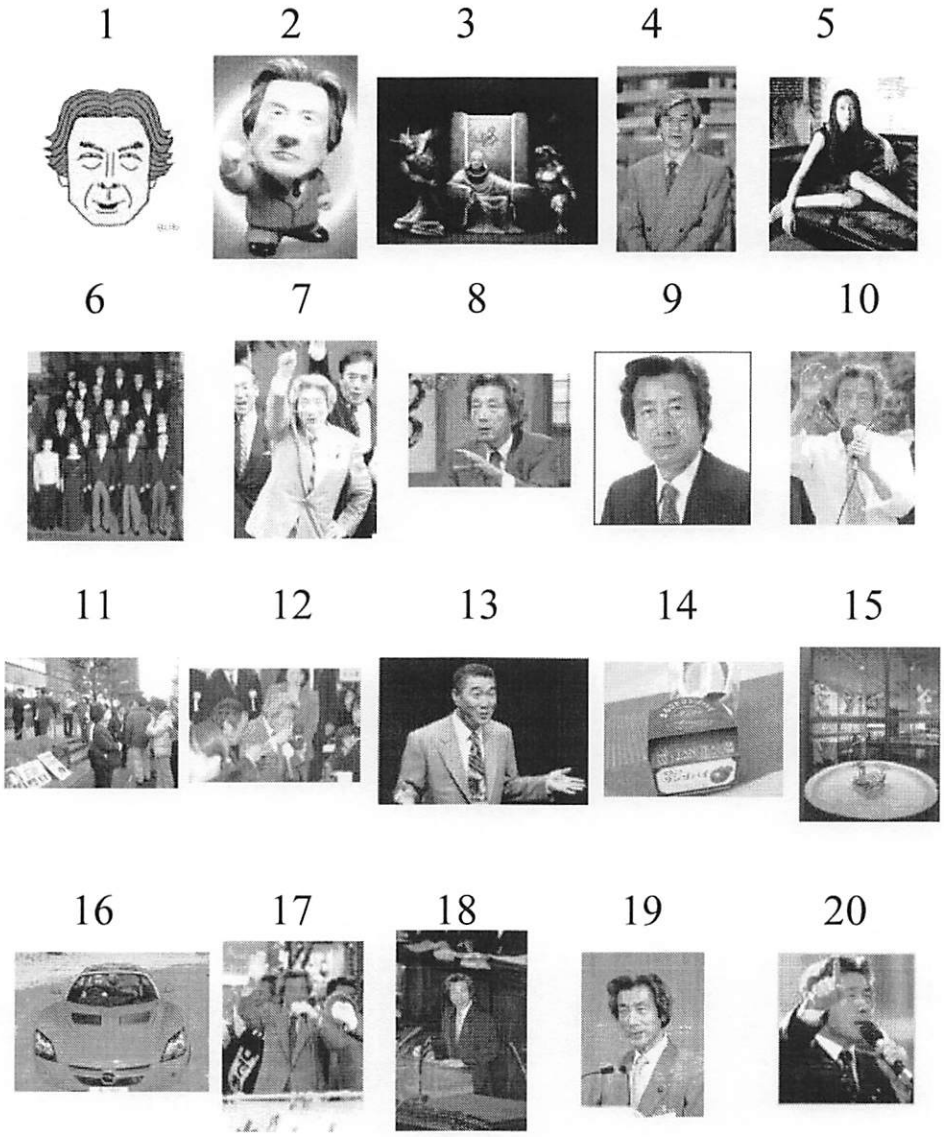


図 4.10: 小泉純一郎の実験データ

図 4.11: 渡辺謙の実験データ

これらのイメージ検索結果に対し本実験を施した後の結果を、図 4.12、図 4.13、図 4.14 にそれぞれ示す。

図 4.12: イチローの ALT 属性に対し重みをつけて行った実験結果

図 4.13: 小泉純一郎の ALT 属性に対し重みをつけて行った実験結果

図 4.14: 渡辺謙の ALT 属性に対し重みをつけて行った実験結果

4.4 検索キーワードを除いた実験

次に 4.3 の実験に加え、Naive Bayes 分類器に入力する訓練データからそれぞれ検索キーワードを除いた実験を行った。これは、「イチロー」の訓練データであればその訓練データを除き、Naive Bayes 分類器に入力を行う実験である。4.2 の各クラスの訓練データである表 4.1、表 4.2、表 4.2 をそれぞれ見てみると、各クラスにおいて最も出現頻度が高いのは検索キーワードである「イチロー」、「小泉純一郎」、「渡辺謙」であることがわかる。特にイチローの訓練データにおいては検索

キーワードである「イチロー」とその他の単語の出現頻度の差が顕著であり、次に多い単語と比べ1.5倍ほどあることがわかる。このことから、Webページの全体的な文章の内容が関連性がかなり低い文章内容でも、「イチロー」などの検索キーワードが出現していると、その出現頻度の高さから、Naive Bayes分類器によって関連性が高い文章であると認識されてしまう。その例を図4.15に示す。

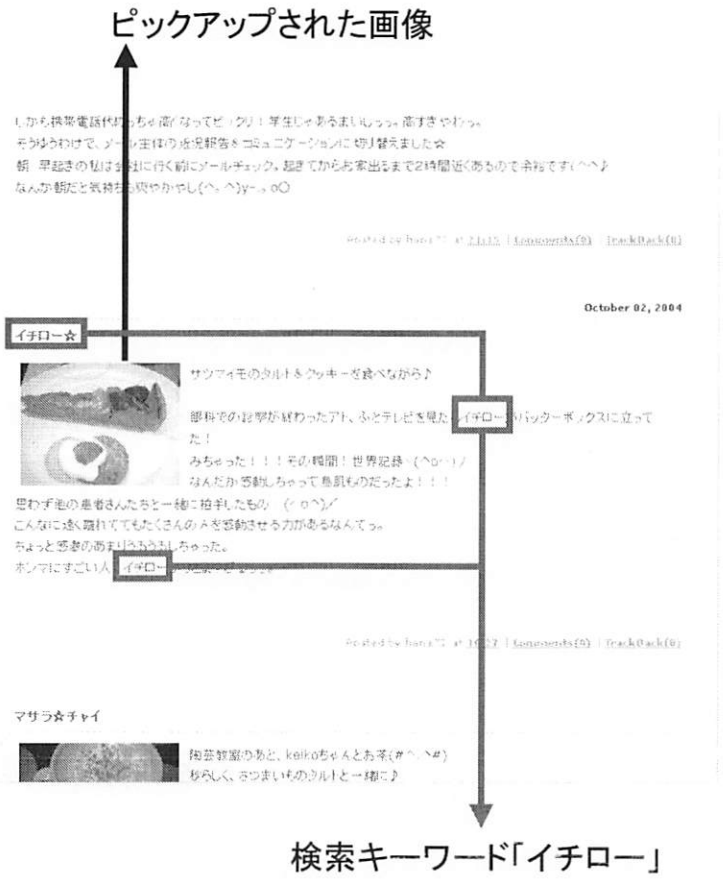


図 4.15: 検索キーワードを含むが関連性が低い画像

このWebページは日付ごとの日記形式となっている。確かにこの画像を含む日付の日記には、検索キーワードである「イチロー」が含まれている。しかし、その内容はイチローに関する分野よりも他の分野のキーワードが多く存在していることがわかる。このことから、訓練データから、出現頻度の高い検索キーワードを取り除くことによって、検索キーワード以外の単語の出現頻度から検索キーワー

ドと Web ページの関連性を計ること出来ると考え実験を行った。

入力する実験データは 4.3 の実験の図 4.9、図 4.10、図 4.11 と同様である。これらの実験データに対し本実験を施した結果を図 4.16、図 4.17、図 4.18 にそれぞれ示す。

図 4.16: イチローの訓練データから検索キーワードを除いた実験結果

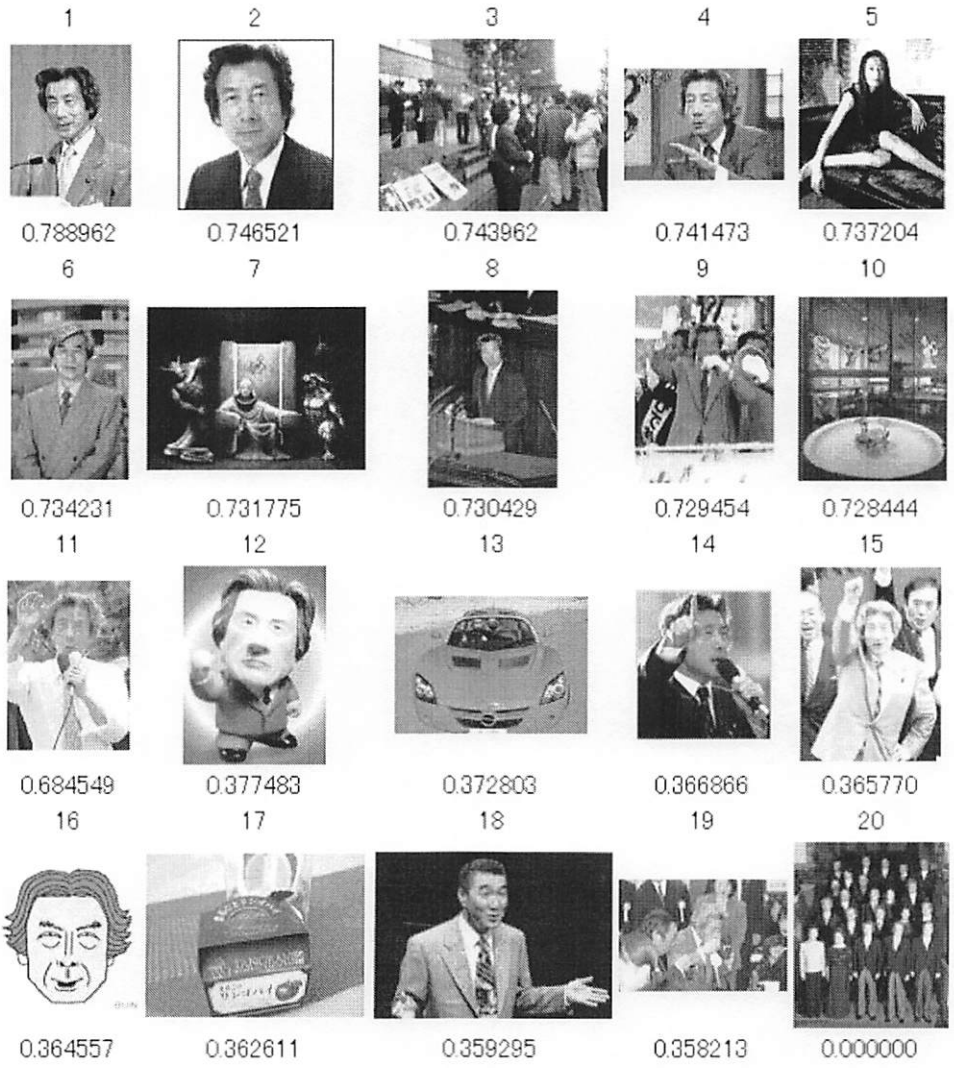


図 4.17: 小泉純一郎の訓練データから検索キーワードを除いた実験結果

図 4.18: 渡辺謙の訓練データから検索キーワードを除いた実験結果

第5章　考察

5.1　重み付け無しの実験についての考察

まずこの実験では、Webページ内の個所によって重み付けを行わない実験を行った。この実験の結果を見ると、Googleのランキングを変更するフィルタリングには成功したと言えるが、当初期待していたような結果を得ることは出来なかった。

この原因として次のようなことが考えられる。まず、Googleイメージの検索結果でサムネイルにピックアップされた画像は、画像元のWebページに含まれる唯一の画像でないことが多い。そのため、画像元のWebページ全体のテキストを見た場合、検索でピックアップされた画像とは異なることまで記述されていることが多々見られた。特に、近年急速に普及しているブログと呼ばれるWeb上で公開する日記のような構造のWebページでではその特徴が顕著に見られた。ブログ上のある日付では確かにピックアップされた画像のジャンルについて記述しているが、翌日の日付では画像と異なる事柄について記述していることがあるなど、分野の一貫性を取っていることは少ない。そのため、ピックアップされた画像が検索キーワードと関連性が高い者であっても、フィルタリングの結果関連性が低いと判断されてしまっていると考えられる。

次に訓練データにおいても原因があると思われる。この実験では訓練データとして、Web上でキーワード検索をした結果からHTML文書を各クラス40件づつ用意した。訓練データを見ると最も出現頻度が高い単語がクラスの人名になっている。しかし、先ほど説明したようなブログが訓練データがに含まれることによって、クラス外の分野の属すると考えられる単語が含まれてしまうデータになってしまった。このため、クラス分類の際に実験データに出現する単語と一致することが多く、それが原因でスコアが上昇する結果となってしまったと言える。

5.2　ALT属性に対し重みをつけて行った実験についての考察

この実験ではHTML文書の画像挿入タグであるIMGタグの属性であるALT属性に着目して実験を行った。またこの実験では、重み付け無しの実験の訓練データの問題点と考えられるクラス外の単語を排除するため、キーワード検索されたWebページの中から、そのWebページの内容を見た結果、検索キーワードについ

て一貫して述べられているWebページのみを訓練データのHTML文書として利用した。

この実験の結果、Googleの検索結果では下位のデータのランキングが上昇してしまっていることが多く見られた。この結果から、ALT属性が画像に直接インデックスをつける形であることから、ある程度良好な結果を得られた部分もあったが、ALT属性の画像と内容の関連性の高さから考えられたような結果を得ることは出来なかった。

この結果の原因として考えられたのは検索キーワードの出現頻度の高さから、実験データのWebページ上で一度でも検索キーワードが出現していると、その出現頻度の高さに引っ張られ、Naive Bayes分類器の分類の結果があがってしまうことが考えられる。

5.3 検索キーワードを除いた実験についての考察

この実験では、ALT属性に対し重みをつけて行う実験において、訓練データから検索キーワードを排除して実験を行った。訓練データにおいて検索キーワードを排除することによって、検索キーワードとの共起する単語によるスコアの算出が出来ると考え、実験を行った。

この実験の結果、上述のALT属性に対し重みをつけてのみ行った実験と比べ、Googleの検索では下位にあった関連性の高い画像を上位にすることが出来た。また、ALT属性に重み付けをして行った実験において上位になってしまっていた関連性が低い画像の順位を、この実験では画像の順位を下位へ変更することが出来た。このことから比較的良好な結果を得ることが出来たと言える。この実験から、やはり検索キーワードが出現するだけで関連性がある画像とは言いがたく、周辺の単語群から検索キーワードの分野を推測する手法は有効である。

第6章　結論

本研究では、キーワードを含む文章を文書分類を用いることにより、誤った検索結果を除外するフィルタ作成に関する研究を行った。今回の研究では、Googleによって提供されている画像検索から、同姓同名の人物の画像など、要求していない画像を排除し、より高精度な画像へと絞込み、目的画像入手までの検索過程の簡略化を行う手法を確立することを目標に研究を行った。実験の結果、比較的良好な結果を得られた例もあったが、一部順位を下げる事のできない画像が未だ存在する、関連性の低い画像のランキングが上昇してしまう、という結果例があった。これらのことから、本研究の目的とする所である、より高精度な検索結果の絞込みを行う手法を確立することができたとは言い難い。

そのため今後の展望として、今回のNaive Bayes法を用いたフィルタリングを基礎とした手法に加え、新たな判断規準を設ける必要がある。現在、その基準1つとして考えてられる手法としては、人名の共起が多いWebページの重み付けを下げると言うことがあげられる。今回の実験において、実験データを検索結果から入手する際、多数の人物の似顔絵の画像を公開しているWebページや、ブログなどにおいて様々な人物について取り上げているページなどがあった。これらのWebページからピックアップされた画像を見てみると、肖像権に違反している恐れがある画像や、キーワードとの関連性が低いと言える画像が多かった。よって、これらのWebページの画像のスコアを低下させることが出来ればさらに良好な結果を得ることが出来ると考えられる。図6.1に画像を含むブログの例を示す。

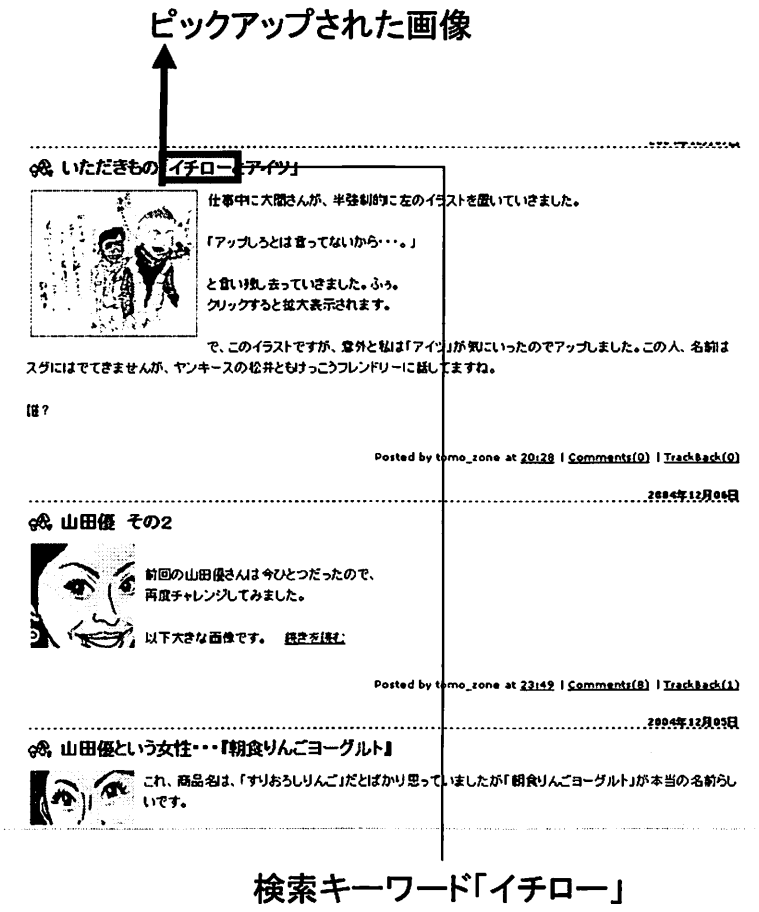


図 6.1: 画像を含むブログ

また、今回は一般的な需要の高さから人物の画像検索に特化したが、Web 上の画像はインターネットの発展と共に、医療や経済などの分野などフレキシブルな利用法が考えられる。このことから、人名以外の様々な分野への対応も必要であると考えられる。

謝辞

本研究の遂行及び論文の作成において多大な助言及び御指導を賜りました新納浩幸教官に深い感謝の意を表します。また副査をしてくださったシステム工学科計算機応用学講座の坪井 一洋教官、情報工学科の佐々木 稔教官に深い感謝いたします。

また、その他様々な助言を頂きました岩崎唯史教官、システム工学科計算機応用学講座の教官の方々、紺野憲一氏、藤井丈明氏、谷津哲平氏、児玉光太郎氏、高橋辰裕氏、木本俊氏、大北高広氏、茂木啓悟氏、藤村元彦氏、高橋宏直氏に深く感謝いたします。

関連図書

[1] Google イメージ検索の研究. http://www.somethingfine.com/br/searchlab/google-image01.html, 2002.

[2] Thomas Michell Tom M.Mitchell. *Machine Learning(Mcgraw-Hill Series in Computer Science)*. McGraw-Hill Science/Engineering/Math.

[3] 谷内田 正彦相良 直樹. Html テキストの重要度を用いた画像ラベリング手法. 人工知能学会全国大会第 17 回, 2003.

[4] 田河 生長他. 確率統計. 大日本図書株式会社.

[5] 佐々木稔, 新納浩幸. 文書分類を用いたスパムメール判定手法. NL163-11. 情報処理学会自然言語処理研究会, September(2004).

付録:A

4.2節の実験の結果と作成プログラムを付録として掲載する.

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

小泉純一郎の画像検索結果上位

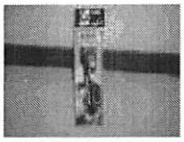
21

22

23

24

25



26

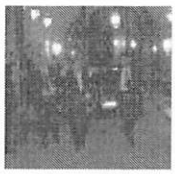
27

28

29

30

31



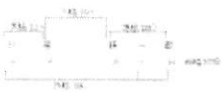
32

33

34

35

36



37

38

39

40

小泉純一郎の画像検索結果中位

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

小泉純一郎の画像検索結果下位

1 2 3 4 5




6 7 8 9 10



11 12 13 14 15





16 17 18 19 20

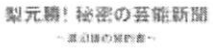


渡辺謙の画像検索結果上位

21

22

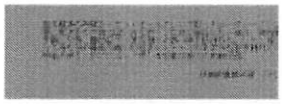
23



24

25

26



27

28

29

30

31

32

33

34

35

36

37

38

39

40

渡辺謙の画像検索結果中位

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

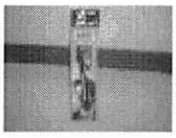
57

58

59

60

渡辺謙の画像検索結果下位

1	2	3	4	5
				
0.405974	0.405974	0.402037	0.401981	0.401021
6	7	8	9	10
				
0.400612	0.398493	0.398459	0.398088	nan
11	12	13	14	15
				
0.397237	0.396965	0.396944	0.395388	0.395232
16	17	18	19	20
				
0.395069	0.393936	0.393919	0.393399	0.393399

小泉純一郎の重み付け無しフィルタリング実験結果1

21

0.393199

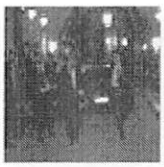
22

0.392669

23

0.390788

24



0.390617

25

0.390514

26

0.390327

27

0.390105

28

0.387458

29

0.387212

30

0.386824

31

0.385126

32

0.384980

33

0.384896

34

0.384646

35

0.382357

36

0.381462

37

0.379591

38

0.378893

39

0.375975

40

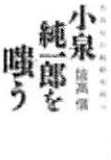
0.375456

小泉純一郎の重み付け無しフィルタリング実験結果2

41

0.374480

42



0.373604

43

0.371750

44

0.371649

45

0.371529

46

0.371529

47

0.369776

48

0.366754

49

0.366369

50

0.366226

51

0.360585

52

0.354084

53

0.347087

54

0.344730

55

0.341922

56

0.296977

57

0.293865

58

0.243709

59

0.233511

60

0.000000

小泉純一郎の重み付け無しフィルタリング実験結果3

1 2 3 4 5




0.164891 0.163017 0.160767 0.160658 0.159607

6 7 8 9 10





0.159367 0.159367 0.159122 0.159090 0.159069

11 12 13 14 15





0.158850 0.158705 0.158646 0.158574 0.158533

16 17 18 19 20





0.158184 0.158053 0.158053 0.157969 0.157762

渡辺謙の重み付け無しフィルタリング実験結果1

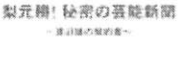
21	22	23	24	25
				
0.157570	0.157321	0.157258	0.157090	0.157050
26	27	28	29	30
				
0.156969	0.156610	0.156580	0.155986	0.155986
31	32	33	34	35
				
0.155986	0.155716	0.155295	0.154814	0.153276
36	37	38	39	40
				
0.151354	0.151260	0.151050	0.150084	0.149674

渡辺謙の重み付け無しフィルタリング実験結果2

41

0.149663

42



0.148416

43

0.148322

44

0.148322

45

0.148322

46

0.145857

47

0.145723

48

0.144291

49

0.142624

50

0.140696

51

0.140244

52

0.137944

53

0.133774

54

0.105243

55

0.000000

56

0.000000

57

0.000000

58

0.000000

59

0.000000

60

0.000000

渡辺謙の重み付け無しフィルタリング実験結果3

付録:B

今回作成したプログラムリストを付録:B として添付する。

4.2 節で使用した訓練データを作成するシェルプログラム

```
for filename in koizumi ichiro watanabe
do
rm -r TRAINING_DATA2/$filename/juman
rm -r TRAINING_DATA2/$filename/C
rm -r TRAINING_DATA2/$filename/sort
rm -r TRAINING_DATA2/$filename/uniq
rm -r TRAINING_DATA2/$filename/results
rm -r TRAINING_DATA2/$filename/EUC
rm -r TRAINING_DATA2/$filename/BUF
rm -r TRAINING_DATA2/$filename/TEXT
rm -r TRAINING_DATA2/$filename/IMG
rm -r TRAINING_DATA2/$filename/ALT
rm -r TRAINING_DATA2/$filename/RESULT

mkdir TRAINING_DATA2/$filename/juman
mkdir TRAINING_DATA2/$filename/C
mkdir TRAINING_DATA2/$filename/sort
mkdir TRAINING_DATA2/$filename/uniq
mkdir TRAINING_DATA2/$filename/results
mkdir TRAINING_DATA2/$filename/EUC
mkdir TRAINING_DATA2/$filename/BUF
mkdir TRAINING_DATA2/$filename/TEXT
mkdir TRAINING_DATA2/$filename/IMG
mkdir TRAINING_DATA2/$filename/ALT
mkdir TRAINING_DATA2/$filename/RESULT

for n in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24  \\
25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
do
gcc imgMODI5.c
if [ -e "TRAINING_DATA2/$filename/HTML/$n.html" ];
then
nkf -e TRAINING_DATA2/$filename/HTML/$n.html >  \\
TRAINING_DATA2/$filename/EUC/EUC$n.html
./a.out TRAINING_DATA2/$filename/EUC/EUC$n.html  \\
TRAINING_DATA2/$filename/BUF/$n.txt  \\
TRAINING_DATA2/$filename/IMG/img$n.txt  \\
TRAINING_DATA2/$filename/ALT/alt$n.txt
nkf -e TRAINING_DATA2/$filename/BUF/$n.txt >  \\
TRAINING_DATA2/$filename/TEXT/$n.txt
fi

gcc exp1.c
juman -e < TRAINING_DATA2/$filename/TEXT/$n.txt >  \\
TRAINING_DATA2/$filename/juman/$n.juman
./a.out TRAINING_DATA2/$filename/juman/$n.juman  \\
TRAINING_DATA2/$filename/C/all$n.txt  \\
TRAINING_DATA2/$filename/C/noun$n.txt  \\
TRAINING_DATA2/$filename/RESULT/ALL.txt  \\
TRAINING_DATA2/$filename/RESULT/NOUN.txt

sort TRAINING_DATA2/$filename/C/all$n.txt >  \\
TRAINING_DATA2/$filename/sort/all$n.txt
uniq -c TRAINING_DATA2/$filename/sort/all$n.txt >  \\
TRAINING_DATA2/$filename/uniq/all$n.txt
sort -r TRAINING_DATA2/$filename/uniq/all$n.txt >  \\
TRAINING_DATA2/$filename/results/all$n.txt

sort TRAINING_DATA2/$filename/C/noun$n.txt >  \\
TRAINING_DATA2/$filename/sort/noun$n.txt
uniq -c TRAINING_DATA2/$filename/sort/noun$n.txt >  \\
TRAINING_DATA2/$filename/uniq/noun$n.txt
sort -r TRAINING_DATA2/$filename/uniq/noun$n.txt >  \\
TRAINING_DATA2/$filename/results/noun$n.txt

sort TRAINING_DATA2/$filename/RESULT/ALL.txt >  \\
```

```
TRAINING_DATA2/$filename/RESULT/SORT.txt
uniq -c TRAINING_DATA2/$filename/RESULT/SORT.txt >  \\
TRAINING_DATA2/$filename/RESULT/COUNT.txt
sort -r TRAINING_DATA2/$filename/RESULT/COUNT.txt >  \\
TRAINING_DATA2/$filename/RESULT/BUF.txt
nkf -e TRAINING_DATA2/$filename/RESULT/BUF.txt >  \\
TRAINING_DATA2/$filename/RESULT/ALL_RESULT.txt

sort TRAINING_DATA2/$filename/RESULT/NOUN.txt >  \\
TRAINING_DATA2/$filename/RESULT/SORT2.txt
uniq -c TRAINING_DATA2/$filename/RESULT/SORT2.txt > \\
 TRAINING_DATA2/$filename/RESULT/COUNT2.txt
sort -r TRAINING_DATA2/$filename/RESULT/COUNT2.txt >  \\
TRAINING_DATA2/$filename/RESULT/BUF2.txt
nkf -e TRAINING_DATA2/$filename/RESULT/BUF2.txt >  \\
TRAINING_DATA2/$filename/RESULT/NOUN_RESULT.txt

echo "$n end"
done
echo "$filename end"
done

for filename in kokusai shakai keizai
do
rm -r TRAINING_DATA2/$filename/juman
rm -r TRAINING_DATA2/$filename/C
rm -r TRAINING_DATA2/$filename/sort
rm -r TRAINING_DATA2/$filename/uniq
rm -r TRAINING_DATA2/$filename/results
rm -r TRAINING_DATA2/$filename/EUC
rm -r TRAINING_DATA2/$filename/BUF
rm -r TRAINING_DATA2/$filename/TEXT
rm -r TRAINING_DATA2/$filename/IMG
rm -r TRAINING_DATA2/$filename/ALT
rm -r TRAINING_DATA2/OTHER

mkdir TRAINING_DATA2/$filename/juman
mkdir TRAINING_DATA2/$filename/C
mkdir TRAINING_DATA2/$filename/sort
mkdir TRAINING_DATA2/$filename/uniq
mkdir TRAINING_DATA2/$filename/results
mkdir TRAINING_DATA2/$filename/EUC
mkdir TRAINING_DATA2/$filename/BUF
mkdir TRAINING_DATA2/$filename/TEXT
mkdir TRAINING_DATA2/$filename/IMG
mkdir TRAINING_DATA2/$filename/ALT
mkdir TRAINING_DATA2/OTHER
for n in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20
do
gcc imgMODI5.c
if [ -e "TRAINING_DATA2/$filename/HTML/$n.html" ];
then
nkf -e TRAINING_DATA2/$filename/HTML/$n.html >  \\
TRAINING_DATA2/$filename/EUC/EUC$n.html
./a.out TRAINING_DATA2/$filename/EUC/EUC$n.html  \\
TRAINING_DATA2/$filename/BUF/$n.txt  \\
TRAINING_DATA2/$filename/IMG/img$n.txt  \\
TRAINING_DATA2/$filename/ALT/alt$n.txt
nkf -e TRAINING_DATA2/$filename/BUF/$n.txt >  \\
TRAINING_DATA2/$filename/TEXT/$n.txt
fi

gcc exp1.c
juman -e < TRAINING_DATA2/$filename/TEXT/$n.txt >  \\
TRAINING_DATA2/$filename/juman/$n.juman
./a.out TRAINING_DATA2/$filename/juman/$n.juman  \\
TRAINING_DATA2/$filename/C/all$n.txt  \\
```

```
TRAINING_DATA2/$filename/C/noun$n.txt  \\
TRAINING_DATA2/OTHER/ALL.txt TRAINING_DATA2/OTHER/NOUN.txt

sort TRAINING_DATA2/$filename/C/all$n.txt >  \\
TRAINING_DATA2/$filename/sort/all$n.txt
uniq -c TRAINING_DATA2/$filename/sort/all$n.txt >  \\
TRAINING_DATA2/$filename/uniq/all$n.txt
sort -r TRAINING_DATA2/$filename/uniq/all$n.txt >  \\
TRAINING_DATA2/$filename/results/all$n.txt

sort TRAINING_DATA2/$filename/C/noun$n.txt >  \\
TRAINING_DATA2/$filename/sort/noun$n.txt
uniq -c TRAINING_DATA2/$filename/sort/noun$n.txt >  \\
TRAINING_DATA2/$filename/uniq/noun$n.txt
sort -r TRAINING_DATA2/$filename/uniq/noun$n.txt >  \\
TRAINING_DATA2/$filename/results/noun$n.txt

done


echo "$filename end"

done

sort TRAINING_DATA2/OTHER/ALL.txt > TRAINING_DATA2/OTHER/SORT.txt
uniq -c TRAINING_DATA2/OTHER/SORT.txt > TRAINING_DATA2/OTHER/COUNT.txt
sort -r TRAINING_DATA2/OTHER/COUNT.txt > TRAINING_DATA2/OTHER/BUF.txt
nkf -e TRAINING_DATA2/OTHER/BUF.txt > TRAINING_DATA2/OTHER/ALL_RESULT.txt


sort TRAINING_DATA2/OTHER/NOUN.txt > TRAINING_DATA2/OTHER/SORT2.txt
uniq -c TRAINING_DATA2/OTHER/SORT2.txt > TRAINING_DATA2/OTHER/COUNT2.txt
sort -r TRAINING_DATA2/OTHER/COUNT2.txt > TRAINING_DATA2/OTHER/BUF2.txt
nkf -e TRAINING_DATA2/OTHER/BUF2.txt > TRAINING_DATA2/OTHER/NOUN_RESULT.txt
```

4.2 節の分類器に入力する実験データを作成するシェルプログラム

```
for filename in koizumi ichiro watanabe
do
#各フォルダ作成
rm -r EXP_DATA4/$filename/juman
rm -r EXP_DATA4/$filename/C
rm -r EXP_DATA4/$filename/sort
rm -r EXP_DATA4/$filename/uniq
rm -r EXP_DATA4/$filename/results
rm -r EXP_DATA4/$filename/EUC
rm -r EXP_DATA4/$filename/BUF
rm -r EXP_DATA4/$filename/TEXT
rm -r EXP_DATA4/$filename/IMG
rm -r EXP_DATA4/$filename/IMG_juman
rm -r EXP_DATA4/$filename/IMG_C
rm -r EXP_DATA4/$filename/IMG_sort
rm -r EXP_DATA4/$filename/IMG_uniq
rm -r EXP_DATA4/$filename/IMG_results
rm -r EXP_DATA4/$filename/ALT
rm -r EXP_DATA4/$filename/ALT_juman
rm -r EXP_DATA4/$filename/ALT_C
rm -r EXP_DATA4/$filename/ALT_sort
rm -r EXP_DATA4/$filename/ALT_uniq
rm -r EXP_DATA4/$filename/ALT_results

mkdir EXP_DATA4/$filename/juman
mkdir EXP_DATA4/$filename/C
mkdir EXP_DATA4/$filename/sort
mkdir EXP_DATA4/$filename/uniq
mkdir EXP_DATA4/$filename/results
mkdir EXP_DATA4/$filename/EUC
mkdir EXP_DATA4/$filename/BUF
mkdir EXP_DATA4/$filename/TEXT
mkdir EXP_DATA4/$filename/IMG
mkdir EXP_DATA4/$filename/IMG_juman
mkdir EXP_DATA4/$filename/IMG_C
mkdir EXP_DATA4/$filename/IMG_sort
mkdir EXP_DATA4/$filename/IMG_uniq
mkdir EXP_DATA4/$filename/IMG_results
mkdir EXP_DATA4/$filename/ALT
mkdir EXP_DATA4/$filename/ALT_juman
mkdir EXP_DATA4/$filename/ALT_C
mkdir EXP_DATA4/$filename/ALT_sort
mkdir EXP_DATA4/$filename/ALT_uniq
mkdir EXP_DATA4/$filename/ALT_results

#上位
for n in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20
do

#HTML からの img,alt, 本文の抽出
gcc imgMODI5.c
if [ -e "EXP_DATA4/$filename/HTML/TOP/$n.html" ];
then
nkf -e EXP_DATA4/$filename/HTML/TOP/$n.html >  \\
EXP_DATA4/$filename/EUC/TOP_EUC$n.html
./a.out EXP_DATA4/$filename/EUC/TOP_EUC$n.html  \\
EXP_DATA4/$filename/BUF/TOP_$n.txt  \\
EXP_DATA4/$filename/IMG/TOP_img$n.txt  \\
EXP_DATA4/$filename/ALT/TOP_alt$n.txt
nkf -e EXP_DATA4/$filename/BUF/TOP_$n.txt >  \\
EXP_DATA4/$filename/TEXT/TOP_$n.txt
fi

#img,alt, 本文中の名詞、未定義語の抽出
gcc exp1.c
```

```
juman -e < EXP_DATA4/$filename/TEXT/TOP_$n.txt >  \\
EXP_DATA4/$filename/juman/TOP_$n.juman
juman -e < EXP_DATA4/$filename/IMG/TOP_img$n.txt >  \\
EXP_DATA4/$filename/IMG_juman/TOP_img$n.juman
juman -e < EXP_DATA4/$filename/ALT/TOP_alt$n.txt >  \\
EXP_DATA4/$filename/ALT_juman/TOP_alt$n.juman

./a.out EXP_DATA4/$filename/juman/TOP_$n.juman  \\
EXP_DATA4/$filename/C/TOP_all$n.txt  \\
EXP_DATA4/$filename/C/TOP_noun$n.txt EXP_DATA4/$filename/C/TOP_ALL$n.txt \\
 EXP_DATA4/$filename/C/TOP_NOUN$n.txt
./a.out EXP_DATA4/$filename/IMG_juman/TOP_img$n.juman  \\
EXP_DATA4/$filename/IMG_C/TOP_all$n.txt  \\
EXP_DATA4/$filename/IMG_C/TOP_noun$n.txt EXP_DATA4/$filename/ALLimg1.txt  \\
EXP_DATA4/$filename/ALLimg2.txt
./a.out EXP_DATA4/$filename/ALT_juman/TOP_alt$n.juman  \\
EXP_DATA4/$filename/ALT_C/TOP_all$n.txt  \\
EXP_DATA4/$filename/ALT_C/TOP_noun$n.txt  \\
EXP_DATA4/$filename/C/TOP_ALL$n.txt  \\
EXP_DATA4/$filename/ALT_C/TOP_NOUN$n.txt

#本文のソート
sort EXP_DATA4/$filename/C/TOP_all$n.txt >  \\
EXP_DATA4/$filename/sort/TOP_all$n.txt
uniq -c EXP_DATA4/$filename/sort/TOP_all$n.txt >  \\
EXP_DATA4/$filename/uniq/TOP_all$n.txt
sort -r EXP_DATA4/$filename/uniq/TOP_all$n.txt >  \\
EXP_DATA4/$filename/results/TOP_all$n.txt
sort EXP_DATA4/$filename/C/TOP_noun$n.txt >  \\
EXP_DATA4/$filename/sort/TOP_noun$n.txt
uniq -c EXP_DATA4/$filename/sort/TOP_noun$n.txt >  \\
EXP_DATA4/$filename/uniq/TOP_noun$n.txt
sort -r EXP_DATA4/$filename/uniq/TOP_noun$n.txt >  \\
EXP_DATA4/$filename/results/TOP_noun$n.txt

#img タグのソート
sort EXP_DATA4/$filename/IMG_C/TOP_all$n.txt >  \\
EXP_DATA4/$filename/IMG_sort/TOP_all$n.txt
uniq -c EXP_DATA4/$filename/IMG_sort/TOP_all$n.txt >  \\
EXP_DATA4/$filename/IMG_uniq/TOP_all$n.txt
sort -r EXP_DATA4/$filename/IMG_uniq/TOP_all$n.txt >  \\
EXP_DATA4/$filename/IMG_results/TOP_all$n.txt
sort EXP_DATA4/$filename/IMG_C/TOP_noun$n.txt >  \\
EXP_DATA4/$filename/IMG_sort/TOP_noun$n.txt
uniq -c EXP_DATA4/$filename/IMG_sort/TOP_noun$n.txt >  \\
EXP_DATA4/$filename/IMG_uniq/TOP_noun$n.txt
sort -r EXP_DATA4/$filename/IMG_uniq/TOP_noun$n.txt >  \\
EXP_DATA4/$filename/IMG_results/TOP_noun$n.txt

#alt 属性のソート
sort EXP_DATA4/$filename/ALT_C/TOP_all$n.txt > \\
 EXP_DATA4/$filename/ALT_sort/TOP_all$n.txt
uniq -c EXP_DATA4/$filename/ALT_sort/TOP_all$n.txt > \\
 EXP_DATA4/$filename/ALT_uniq/TOP_all$n.txt
sort -r EXP_DATA4/$filename/ALT_uniq/TOP_all$n.txt > \\
 EXP_DATA4/$filename/ALT_results/TOP_all$n.txt
sort EXP_DATA4/$filename/ALT_C/TOP_noun$n.txt >  \\
EXP_DATA4/$filename/ALT_sort/TOP_noun$n.txt
uniq -c EXP_DATA4/$filename/ALT_sort/TOP_noun$n.txt > \\
 EXP_DATA4/$filename/ALT_uniq/TOP_noun$n.txt
sort -r EXP_DATA4/$filename/ALT_uniq/TOP_noun$n.txt >  \\
EXP_DATA4/$filename/ALT_results/TOP_noun$n.txt



#
sort EXP_DATA4/$filename/C/TOP_ALL$n.txt > EXP_DATA4/$filename/sort/TOP_ALL$n.txt
```

```
uniq -c EXP_DATA4/$filename/sort/TOP_ALL$n.txt > EXP_DATA4/$filename/uniq/TOP_ALL$n.txt
sort -r EXP_DATA4/$filename/uniq/TOP_ALL$n.txt > EXP_DATA4/$filename/results/TOP_ALL$n.txt
sort EXP_DATA4/$filename/C/TOP_NOUN$n.txt > EXP_DATA4/$filename/sort/TOP_NOUN$n.txt
uniq -c EXP_DATA4/$filename/sort/TOP_NOUN$n.txt > EXP_DATA4/$filename/uniq/TOP_NOUN$n.txt
sort -r EXP_DATA4/$filename/uniq/TOP_NOUN$n.txt > EXP_DATA4/$filename/results/TOP_NOUN$n.txt

echo "$n end"
done
echo "TOP end"

#中間
for n in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20
do

#HTML からの img,alt, 本文の抽出
gcc imgMODI5.c
if [ -e "EXP_DATA4/$filename/HTML/MID/$n.html" ];
then
nkf -e EXP_DATA4/$filename/HTML/MID/$n.html > EXP_DATA4/$filename/EUC/MID_EUC$n.html
./a.out EXP_DATA4/$filename/EUC/MID_EUC$n.html EXP_DATA4/$filename/BUF/MID_$n.txt  \\
EXP_DATA4/$filename/IMG/MID_img$n.txt EXP_DATA4/$filename/ALT/MID_alt$n.txt
nkf -e EXP_DATA4/$filename/BUF/MID_$n.txt > EXP_DATA4/$filename/TEXT/MID_$n.txt
fi
echo "be"
#img,alt, 本文中の名詞、未定義語の抽出
gcc exp1.c
juman -e < EXP_DATA4/$filename/TEXT/MID_$n.txt > EXP_DATA4/$filename/juman/MID_$n.juman
juman -e < EXP_DATA4/$filename/IMG/MID_img$n.txt > EXP_DATA4/$filename/IMG_juman/MID_img$n.juman
juman -e < EXP_DATA4/$filename/ALT/MID_alt$n.txt > EXP_DATA4/$filename/ALT_juman/MID_alt$n.juman

./a.out EXP_DATA4/$filename/juman/MID_$n.juman  \\
EXP_DATA4/$filename/C/MID_all$n.txt  \\
EXP_DATA4/$filename/C/MID_noun$n.txt EXP_DATA4/$filename/C/MID_ALL$n.txt  \\
EXP_DATA4/$filename/C/MID_NOUN$n.txt
./a.out EXP_DATA4/$filename/IMG_juman/MID_img$n.juman \\
 EXP_DATA4/$filename/IMG_C/MID_all$n.txt  \\
EXP_DATA4/$filename/IMG_C/MID_noun$n.txt EXP_DATA4/$filename/ALLimg1.txt \\
 EXP_DATA4/$filename/ALLimg2.txt
./a.out EXP_DATA4/$filename/ALT_juman/MID_alt$n.juman  \\
EXP_DATA4/$filename/ALT_C/MID_all$n.txt  \\
EXP_DATA4/$filename/ALT_C/MID_noun$n.txt EXP_DATA4/$filename/C/MID_ALL$n.txt  \\
EXP_DATA4/$filename/ALT_C/MID_NOUN$n.txt

#本文のソート
sort EXP_DATA4/$filename/C/MID_all$n.txt >  \\
EXP_DATA4/$filename/sort/MID_all$n.txt
uniq -c EXP_DATA4/$filename/sort/MID_all$n.txt > \\
 EXP_DATA4/$filename/uniq/MID_all$n.txt
sort -r EXP_DATA4/$filename/uniq/MID_all$n.txt > \\
 EXP_DATA4/$filename/results/MID_all$n.txt
sort EXP_DATA4/$filename/C/MID_noun$n.txt >  \\
EXP_DATA4/$filename/sort/MID_noun$n.txt
uniq -c EXP_DATA4/$filename/sort/MID_noun$n.txt >  \\
EXP_DATA4/$filename/uniq/MID_noun$n.txt
sort -r EXP_DATA4/$filename/uniq/MID_noun$n.txt > \\
 EXP_DATA4/$filename/results/MID_noun$n.txt

#img タグのソート
sort EXP_DATA4/$filename/IMG_C/MID_all$n.txt >  \\
EXP_DATA4/$filename/IMG_sort/MID_all$n.txt
uniq -c EXP_DATA4/$filename/IMG_sort/MID_all$n.txt >  \\
EXP_DATA4/$filename/IMG_uniq/MID_all$n.txt
sort -r EXP_DATA4/$filename/IMG_uniq/MID_all$n.txt >  \\
EXP_DATA4/$filename/IMG_results/MID_all$n.txt
sort EXP_DATA4/$filename/IMG_C/MID_noun$n.txt >  \\
EXP_DATA4/$filename/IMG_sort/MID_noun$n.txt
uniq -c EXP_DATA4/$filename/IMG_sort/MID_noun$n.txt > \\
```

```
 EXP_DATA4/$filename/IMG_uniq/MID_noun$n.txt
sort -r EXP_DATA4/$filename/IMG_uniq/MID_noun$n.txt >  \\
EXP_DATA4/$filename/IMG_results/MID_noun$n.txt

#alt 属性のソート
sort EXP_DATA4/$filename/ALT_C/MID_all$n.txt >  \\
EXP_DATA4/$filename/ALT_sort/MID_all$n.txt
uniq -c EXP_DATA4/$filename/ALT_sort/MID_all$n.txt > \\
 EXP_DATA4/$filename/ALT_uniq/MID_all$n.txt
sort -r EXP_DATA4/$filename/ALT_uniq/MID_all$n.txt >  \\
EXP_DATA4/$filename/ALT_results/MID_all$n.txt
sort EXP_DATA4/$filename/ALT_C/MID_noun$n.txt >  \\
EXP_DATA4/$filename/ALT_sort/MID_noun$n.txt
uniq -c EXP_DATA4/$filename/ALT_sort/MID_noun$n.txt >  \\
EXP_DATA4/$filename/ALT_uniq/MID_noun$n.txt
sort -r EXP_DATA4/$filename/ALT_uniq/MID_noun$n.txt >  \\
EXP_DATA4/$filename/ALT_results/MID_noun$n.txt

#
sort EXP_DATA4/$filename/C/MID_ALL$n.txt >  \\
EXP_DATA4/$filename/sort/MID_ALL$n.txt
uniq -c EXP_DATA4/$filename/sort/MID_ALL$n.txt >  \\
EXP_DATA4/$filename/uniq/MID_ALL$n.txt
sort -r EXP_DATA4/$filename/uniq/MID_ALL$n.txt >  \\
EXP_DATA4/$filename/results/MID_ALL$n.txt
sort EXP_DATA4/$filename/C/MID_NOUN$n.txt >  \\
EXP_DATA4/$filename/sort/MID_NOUN$n.txt
uniq -c EXP_DATA4/$filename/sort/MID_NOUN$n.txt > \\
 EXP_DATA4/$filename/uniq/MID_NOUN$n.txt
sort -r EXP_DATA4/$filename/uniq/MID_NOUN$n.txt >  \\
EXP_DATA4/$filename/results/MID_NOUN$n.txt

echo "$n end"
done
echo "MID end"

#下位
for n in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20
do

#HTML からの img,alt, 本文の抽出
gcc imgMODI5.c
if [ -e "EXP_DATA4/$filename/HTML/LAST/$n.html" ];
then
nkf -e EXP_DATA4/$filename/HTML/LAST/$n.html >  \\
EXP_DATA4/$filename/EUC/LAST_EUC$n.html
./a.out EXP_DATA4/$filename/EUC/LAST_EUC$n.html  \\
EXP_DATA4/$filename/BUF/LAST_$n.txt  \\
EXP_DATA4/$filename/IMG/LAST_img$n.txt  \\
EXP_DATA4/$filename/ALT/LAST_alt$n.txt
nkf -e EXP_DATA4/$filename/BUF/LAST_$n.txt >  \\
EXP_DATA4/$filename/TEXT/LAST_$n.txt
fi

#img,alt, 本文中の名詞、未定義語の抽出
gcc exp1.c
juman -e < EXP_DATA4/$filename/TEXT/LAST_$n.txt >  \\
EXP_DATA4/$filename/juman/LAST_$n.juman
juman -e < EXP_DATA4/$filename/IMG/LAST_img$n.txt > \\
 EXP_DATA4/$filename/IMG_juman/LAST_img$n.juman
juman -e < EXP_DATA4/$filename/ALT/LAST_alt$n.txt >  \\
EXP_DATA4/$filename/ALT_juman/LAST_alt$n.juman

./a.out EXP_DATA4/$filename/juman/LAST_$n.juman  \\
EXP_DATA4/$filename/C/LAST_all$n.txt  \\
EXP_DATA4/$filename/C/LAST_noun$n.txt EXP_DATA4/$filename/C/LAST_ALL$n.txt  \\
EXP_DATA4/$filename/C/LAST_NOUN$n.txt
```

```
./a.out EXP_DATA4/$filename/IMG_juman/LAST_img$n.juman  \\
EXP_DATA4/$filename/IMG_C/LAST_all$n.txt  \\
EXP_DATA4/$filename/IMG_C/LAST_noun$n.txt EXP_DATA4/$filename/ALLimg1.txt \\
 EXP_DATA4/$filename/ALLimg2.txt
./a.out EXP_DATA4/$filename/ALT_juman/LAST_alt$n.juman  \\
EXP_DATA4/$filename/ALT_C/LAST_all$n.txt  \\
EXP_DATA4/$filename/ALT_C/LAST_noun$n.txt EXP_DATA4/$filename/C/LAST_ALL$n.txt  \\
EXP_DATA4/$filename/ALT_C/LAST_NOUN$n.txt

#本文のソート
sort EXP_DATA4/$filename/C/LAST_all$n.txt >  \\
EXP_DATA4/$filename/sort/LAST_all$n.txt
uniq -c EXP_DATA4/$filename/sort/LAST_all$n.txt >  \\
EXP_DATA4/$filename/uniq/LAST_all$n.txt
sort -r EXP_DATA4/$filename/uniq/LAST_all$n.txt >  \\
EXP_DATA4/$filename/results/LAST_all$n.txt
sort EXP_DATA4/$filename/C/LAST_noun$n.txt >  \\
EXP_DATA4/$filename/sort/LAST_noun$n.txt
uniq -c EXP_DATA4/$filename/sort/LAST_noun$n.txt >  \\
EXP_DATA4/$filename/uniq/LAST_noun$n.txt
sort -r EXP_DATA4/$filename/uniq/LAST_noun$n.txt >  \\
EXP_DATA4/$filename/results/LAST_noun$n.txt

#img タグのソート
sort EXP_DATA4/$filename/IMG_C/LAST_all$n.txt >  \\
EXP_DATA4/$filename/IMG_sort/LAST_all$n.txt
uniq -c EXP_DATA4/$filename/IMG_sort/LAST_all$n.txt > \\
 EXP_DATA4/$filename/IMG_uniq/LAST_all$n.txt
sort -r EXP_DATA4/$filename/IMG_uniq/LAST_all$n.txt >  \\
EXP_DATA4/$filename/IMG_results/LAST_all$n.txt
sort EXP_DATA4/$filename/IMG_C/LAST_noun$n.txt >  \\
EXP_DATA4/$filename/IMG_sort/LAST_noun$n.txt
uniq -c EXP_DATA4/$filename/IMG_sort/LAST_noun$n.txt > \\
 EXP_DATA4/$filename/IMG_uniq/LAST_noun$n.txt
sort -r EXP_DATA4/$filename/IMG_uniq/LAST_noun$n.txt >  \\
EXP_DATA4/$filename/IMG_results/LAST_noun$n.txt

#alt 属性のソート
sort EXP_DATA4/$filename/ALT_C/LAST_all$n.txt >  \\
EXP_DATA4/$filename/ALT_sort/LAST_all$n.txt
uniq -c EXP_DATA4/$filename/ALT_sort/LAST_all$n.txt >  \\
EXP_DATA4/$filename/ALT_uniq/LAST_all$n.txt
sort -r EXP_DATA4/$filename/ALT_uniq/LAST_all$n.txt > \\
 EXP_DATA4/$filename/ALT_results/LAST_all$n.txt
sort EXP_DATA4/$filename/ALT_C/LAST_noun$n.txt >  \\
EXP_DATA4/$filename/ALT_sort/LAST_noun$n.txt
uniq -c EXP_DATA4/$filename/ALT_sort/LAST_noun$n.txt > \\
 EXP_DATA4/$filename/ALT_uniq/LAST_noun$n.txt
sort -r EXP_DATA4/$filename/ALT_uniq/LAST_noun$n.txt >  \\
EXP_DATA4/$filename/ALT_results/LAST_noun$n.txt

#
sort EXP_DATA4/$filename/C/LAST_ALL$n.txt > \\
 EXP_DATA4/$filename/sort/LAST_ALL$n.txt
uniq -c EXP_DATA4/$filename/sort/LAST_ALL$n.txt >  \\
EXP_DATA4/$filename/uniq/LAST_ALL$n.txt
sort -r EXP_DATA4/$filename/uniq/LAST_ALL$n.txt >  \\
EXP_DATA4/$filename/results/LAST_ALL$n.txt
sort EXP_DATA4/$filename/C/LAST_NOUN$n.txt >  \\
EXP_DATA4/$filename/sort/LAST_NOUN$n.txt
uniq -c EXP_DATA4/$filename/sort/LAST_NOUN$n.txt > \\
 EXP_DATA4/$filename/uniq/LAST_NOUN$n.txt
sort -r EXP_DATA4/$filename/uniq/LAST_NOUN$n.txt >  \\
EXP_DATA4/$filename/results/LAST_NOUN$n.txt

echo "$n end"
done
```

```
echo "LAST end"

echo "$filename end"
done
```

4.3 節 4.4 節で使用する訓練データを作成するシェルプログラム

```
for filename in koizumi ichiro watanabe
do
rm -r TRAINING_DATA4/$filename/juman
rm -r TRAINING_DATA4/$filename/C
rm -r TRAINING_DATA4/$filename/sort
rm -r TRAINING_DATA4/$filename/uniq
rm -r TRAINING_DATA4/$filename/results
rm -r TRAINING_DATA4/$filename/EUC
rm -r TRAINING_DATA4/$filename/BUF
rm -r TRAINING_DATA4/$filename/TEXT
rm -r TRAINING_DATA4/$filename/IMG
rm -r TRAINING_DATA4/$filename/ALT
rm -r TRAINING_DATA4/$filename/RESULT

mkdir TRAINING_DATA4/$filename/juman
mkdir TRAINING_DATA4/$filename/C
mkdir TRAINING_DATA4/$filename/sort
mkdir TRAINING_DATA4/$filename/uniq
mkdir TRAINING_DATA4/$filename/results
mkdir TRAINING_DATA4/$filename/EUC
mkdir TRAINING_DATA4/$filename/BUF
mkdir TRAINING_DATA4/$filename/TEXT
mkdir TRAINING_DATA4/$filename/IMG
mkdir TRAINING_DATA4/$filename/ALT
mkdir TRAINING_DATA4/$filename/RESULT

for n in 01 02 03 04 05
do
gcc imgMODI5.c
if [ -e "TRAINING_DATA4/$filename/HTML/$n.html" ];
then
nkf -e TRAINING_DATA4/$filename/HTML/$n.html >  \\
TRAINING_DATA4/$filename/EUC/EUC$n.html
./a.out TRAINING_DATA4/$filename/EUC/EUC$n.html  \\
TRAINING_DATA4/$filename/BUF/$n.txt  \\
TRAINING_DATA4/$filename/IMG/img$n.txt  \\
TRAINING_DATA4/$filename/ALT/alt$n.txt
nkf -e TRAINING_DATA4/$filename/BUF/$n.txt >  \\
TRAINING_DATA4/$filename/TEXT/$n.txt
fi

gcc exp1.c \\

juman -e < TRAINING_DATA4/$filename/TEXT/$n.txt >  \\
TRAINING_DATA4/$filename/juman/$n.juman
./a.out TRAINING_DATA4/$filename/juman/$n.juman  \\
TRAINING_DATA4/$filename/C/all$n.txt  \\
TRAINING_DATA4/$filename/C/noun$n.txt \\
 TRAINING_DATA4/$filename/RESULT/ALL.txt  \\
TRAINING_DATA4/$filename/RESULT/NOUN.txt

sort TRAINING_DATA4/$filename/C/all$n.txt >  \\
TRAINING_DATA4/$filename/sort/all$n.txt
uniq -c TRAINING_DATA4/$filename/sort/all$n.txt > \\
 TRAINING_DATA4/$filename/uniq/all$n.txt
sort -r TRAINING_DATA4/$filename/uniq/all$n.txt >  \\
TRAINING_DATA4/$filename/results/all$n.txt

sort TRAINING_DATA4/$filename/C/noun$n.txt > \\
 TRAINING_DATA4/$filename/sort/noun$n.txt
uniq -c TRAINING_DATA4/$filename/sort/noun$n.txt > \\
 TRAINING_DATA4/$filename/uniq/noun$n.txt
sort -r TRAINING_DATA4/$filename/uniq/noun$n.txt > \\
 TRAINING_DATA4/$filename/results/noun$n.txt
```

```
sort TRAINING_DATA4/$filename/RESULT/ALL.txt > \\
 TRAINING_DATA4/$filename/RESULT/SORT.txt
uniq -c TRAINING_DATA4/$filename/RESULT/SORT.txt >  \\
TRAINING_DATA4/$filename/RESULT/COUNT.txt
sort -r TRAINING_DATA4/$filename/RESULT/COUNT.txt >  \\
TRAINING_DATA4/$filename/RESULT/BUF.txt
nkf -e TRAINING_DATA4/$filename/RESULT/BUF.txt > \\
 TRAINING_DATA4/$filename/RESULT/ALL_RESULT.txt

sort TRAINING_DATA4/$filename/RESULT/NOUN.txt > \\
 TRAINING_DATA4/$filename/RESULT/SORT2.txt
uniq -c TRAINING_DATA4/$filename/RESULT/SORT2.txt >  \\
TRAINING_DATA4/$filename/RESULT/COUNT2.txt
sort -r TRAINING_DATA4/$filename/RESULT/COUNT2.txt > \\
 TRAINING_DATA4/$filename/RESULT/BUF2.txt
nkf -e TRAINING_DATA4/$filename/RESULT/BUF2.txt > \\
 TRAINING_DATA4/$filename/RESULT/NOUN_RESULT.txt

echo "$n end"
done
echo "$filename end"
done

for filename in kokusai shakai keizai
do
rm -r TRAINING_DATA4/$filename/juman
rm -r TRAINING_DATA4/$filename/C
rm -r TRAINING_DATA4/$filename/sort
rm -r TRAINING_DATA4/$filename/uniq
rm -r TRAINING_DATA4/$filename/results
rm -r TRAINING_DATA4/$filename/EUC
rm -r TRAINING_DATA4/$filename/BUF
rm -r TRAINING_DATA4/$filename/TEXT
rm -r TRAINING_DATA4/$filename/IMG
rm -r TRAINING_DATA4/$filename/ALT
rm -r TRAINING_DATA4/OTHER

mkdir TRAINING_DATA4/$filename/juman
mkdir TRAINING_DATA4/$filename/C
mkdir TRAINING_DATA4/$filename/sort
mkdir TRAINING_DATA4/$filename/uniq
mkdir TRAINING_DATA4/$filename/results
mkdir TRAINING_DATA4/$filename/EUC
mkdir TRAINING_DATA4/$filename/BUF
mkdir TRAINING_DATA4/$filename/TEXT
mkdir TRAINING_DATA4/$filename/IMG
mkdir TRAINING_DATA4/$filename/ALT
mkdir TRAINING_DATA4/OTHER
for n in 01 02 03 04 05
do
gcc imgMODI5.c
if [ -e "TRAINING_DATA4/$filename/HTML/$n.html" ];
then
nkf -e TRAINING_DATA4/$filename/HTML/$n.html >  \\
TRAINING_DATA4/$filename/EUC/EUC$n.html
./a.out TRAINING_DATA4/$filename/EUC/EUC$n.html  \\
TRAINING_DATA4/$filename/BUF/$n.txt  \\
TRAINING_DATA4/$filename/IMG/img$n.txt  \\
TRAINING_DATA4/$filename/ALT/alt$n.txt
nkf -e TRAINING_DATA4/$filename/BUF/$n.txt >  \\
TRAINING_DATA4/$filename/TEXT/$n.txt
fi

gcc exp1.c
juman -e < TRAINING_DATA4/$filename/TEXT/$n.txt >  \\
TRAINING_DATA4/$filename/juman/$n.juman
./a.out TRAINING_DATA4/$filename/juman/$n.juman  \\
```

```
TRAINING_DATA4/$filename/C/all$n.txt  \\
TRAINING_DATA4/$filename/C/noun$n.txt  \\
TRAINING_DATA4/OTHER/ALL.txt TRAINING_DATA4/OTHER/NOUN.txt

sort TRAINING_DATA4/$filename/C/all$n.txt >  \\
TRAINING_DATA4/$filename/sort/all$n.txt
uniq -c TRAINING_DATA4/$filename/sort/all$n.txt > \\
 TRAINING_DATA4/$filename/uniq/all$n.txt
sort -r TRAINING_DATA4/$filename/uniq/all$n.txt > \\
 TRAINING_DATA4/$filename/results/all$n.txt

sort TRAINING_DATA4/$filename/C/noun$n.txt >  \\
TRAINING_DATA4/$filename/sort/noun$n.txt
uniq -c TRAINING_DATA4/$filename/sort/noun$n.txt > \\
 TRAINING_DATA4/$filename/uniq/noun$n.txt
sort -r TRAINING_DATA4/$filename/uniq/noun$n.txt > \\
 TRAINING_DATA4/$filename/results/noun$n.txt

done


echo "$filename end"

done

sort TRAINING_DATA4/OTHER/ALL.txt > TRAINING_DATA4/OTHER/SORT.txt
uniq -c TRAINING_DATA4/OTHER/SORT.txt > TRAINING_DATA4/OTHER/COUNT.txt
sort -r TRAINING_DATA4/OTHER/COUNT.txt > TRAINING_DATA4/OTHER/BUF.txt
nkf -e TRAINING_DATA4/OTHER/BUF.txt > TRAINING_DATA4/OTHER/ALL_RESULT.txt


sort TRAINING_DATA4/OTHER/NOUN.txt > TRAINING_DATA4/OTHER/SORT2.txt
uniq -c TRAINING_DATA4/OTHER/SORT2.txt > TRAINING_DATA4/OTHER/COUNT2.txt
sort -r TRAINING_DATA4/OTHER/COUNT2.txt > TRAINING_DATA4/OTHER/BUF2.txt
nkf -e TRAINING_DATA4/OTHER/BUF2.txt > TRAINING_DATA4/OTHER/NOUN_RESULT.txt
```

4.3 節 4.4 節で使用する実験データを作成するシェルプログラム

```
for filename in koizumi ichiro watanabe
do
#各フォルダ作成
rm -r EXP_DATA5/$filename/juman
rm -r EXP_DATA5/$filename/C
rm -r EXP_DATA5/$filename/sort
rm -r EXP_DATA5/$filename/uniq
rm -r EXP_DATA5/$filename/results
rm -r EXP_DATA5/$filename/EUC
rm -r EXP_DATA5/$filename/BUF
rm -r EXP_DATA5/$filename/TEXT
rm -r EXP_DATA5/$filename/IMG
rm -r EXP_DATA5/$filename/IMG_juman
rm -r EXP_DATA5/$filename/IMG_C
rm -r EXP_DATA5/$filename/IMG_sort
rm -r EXP_DATA5/$filename/IMG_uniq
rm -r EXP_DATA5/$filename/IMG_results
rm -r EXP_DATA5/$filename/ALT
rm -r EXP_DATA5/$filename/ALT_juman
rm -r EXP_DATA5/$filename/ALT_C
rm -r EXP_DATA5/$filename/ALT_sort
rm -r EXP_DATA5/$filename/ALT_uniq
rm -r EXP_DATA5/$filename/ALT_results

mkdir EXP_DATA5/$filename/juman
mkdir EXP_DATA5/$filename/C
mkdir EXP_DATA5/$filename/sort
mkdir EXP_DATA5/$filename/uniq
mkdir EXP_DATA5/$filename/results
mkdir EXP_DATA5/$filename/EUC
mkdir EXP_DATA5/$filename/BUF
mkdir EXP_DATA5/$filename/TEXT
mkdir EXP_DATA5/$filename/IMG
mkdir EXP_DATA5/$filename/IMG_juman
mkdir EXP_DATA5/$filename/IMG_C
mkdir EXP_DATA5/$filename/IMG_sort
mkdir EXP_DATA5/$filename/IMG_uniq
mkdir EXP_DATA5/$filename/IMG_results
mkdir EXP_DATA5/$filename/ALT
mkdir EXP_DATA5/$filename/ALT_juman
mkdir EXP_DATA5/$filename/ALT_C
mkdir EXP_DATA5/$filename/ALT_sort
mkdir EXP_DATA5/$filename/ALT_uniq
mkdir EXP_DATA5/$filename/ALT_results

#上位
for n in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20
do

#HTML からの img,alt, 本文の抽出
gcc imgMODI5.c
if [ -e "EXP_DATA5/$filename/HTML/$n.html" ];
then
nkf -e EXP_DATA5/$filename/HTML/$n.html > EXP_DATA5/$filename/EUC/EUC$n.html
./a.out EXP_DATA5/$filename/EUC/EUC$n.html EXP_DATA5/$filename/BUF/$n.txt  \\
EXP_DATA5/$filename/IMG/img$n.txt EXP_DATA5/$filename/ALT/alt$n.txt
nkf -e EXP_DATA5/$filename/BUF/$n.txt > EXP_DATA5/$filename/TEXT/$n.txt
fi
echo "be"
#img,alt, 本文中の名詞、未定義語の抽出
gcc exp1.c
juman -e < EXP_DATA5/$filename/TEXT/$n.txt > \\
 EXP_DATA5/$filename/juman/$n.juman
juman -e < EXP_DATA5/$filename/IMG/img$n.txt > \\
 EXP_DATA5/$filename/IMG_juman/img$n.juman
```

```
juman -e < EXP_DATA5/$filename/ALT/alt$n.txt > \\
 EXP_DATA5/$filename/ALT_juman/alt$n.juman

./a.out EXP_DATA5/$filename/juman/$n.juman \\
 EXP_DATA5/$filename/C/all$n.txt  \\
EXP_DATA5/$filename/C/noun$n.txt EXP_DATA5/$filename/C/ALL$n.txt  \\
EXP_DATA5/$filename/C/NOUN$n.txt
./a.out EXP_DATA5/$filename/IMG_juman/img$n.juman \\
 EXP_DATA5/$filename/IMG_C/all$n.txt  \\
EXP_DATA5/$filename/IMG_C/noun$n.txt EXP_DATA5/$filename/ALLimg1.txt \\
 EXP_DATA5/$filename/ALLimg2.txt
./a.out EXP_DATA5/$filename/ALT_juman/alt$n.juman \\
 EXP_DATA5/$filename/ALT_C/all$n.txt  \\
EXP_DATA5/$filename/ALT_C/noun$n.txt EXP_DATA5/$filename/C/ALL$n.txt \\
 EXP_DATA5/$filename/ALT_C/NOUN$n.txt

#本文のソート
sort EXP_DATA5/$filename/C/all$n.txt > \\
 EXP_DATA5/$filename/sort/all$n.txt
uniq -c EXP_DATA5/$filename/sort/all$n.txt > \\
 EXP_DATA5/$filename/uniq/all$n.txt
sort -r EXP_DATA5/$filename/uniq/all$n.txt >  \\
EXP_DATA5/$filename/results/all$n.txt
sort EXP_DATA5/$filename/C/noun$n.txt >  \\
EXP_DATA5/$filename/sort/noun$n.txt
uniq -c EXP_DATA5/$filename/sort/noun$n.txt > \\
 EXP_DATA5/$filename/uniq/noun$n.txt
sort -r EXP_DATA5/$filename/uniq/noun$n.txt > \\
 EXP_DATA5/$filename/results/noun$n.txt

#img タグのソート
sort EXP_DATA5/$filename/IMG_C/all$n.txt >  \\
EXP_DATA5/$filename/IMG_sort/all$n.txt
uniq -c EXP_DATA5/$filename/IMG_sort/all$n.txt > \\
 EXP_DATA5/$filename/IMG_uniq/all$n.txt
sort -r EXP_DATA5/$filename/IMG_uniq/all$n.txt >  \\
EXP_DATA5/$filename/IMG_results/all$n.txt
sort EXP_DATA5/$filename/IMG_C/noun$n.txt > \\
 EXP_DATA5/$filename/IMG_sort/noun$n.txt
uniq -c EXP_DATA5/$filename/IMG_sort/noun$n.txt > \\
 EXP_DATA5/$filename/IMG_uniq/noun$n.txt
sort -r EXP_DATA5/$filename/IMG_uniq/noun$n.txt > \\
 EXP_DATA5/$filename/IMG_results/noun$n.txt

#alt 属性のソート
sort EXP_DATA5/$filename/ALT_C/all$n.txt >  \\
EXP_DATA5/$filename/ALT_sort/all$n.txt
uniq -c EXP_DATA5/$filename/ALT_sort/all$n.txt >  \\
EXP_DATA5/$filename/ALT_uniq/all$n.txt
sort -r EXP_DATA5/$filename/ALT_uniq/all$n.txt >  \\
EXP_DATA5/$filename/ALT_results/all$n.txt
sort EXP_DATA5/$filename/ALT_C/noun$n.txt >  \\
EXP_DATA5/$filename/ALT_sort/noun$n.txt
uniq -c EXP_DATA5/$filename/ALT_sort/noun$n.txt >  \\
EXP_DATA5/$filename/ALT_uniq/noun$n.txt
sort -r EXP_DATA5/$filename/ALT_uniq/noun$n.txt > \\
 EXP_DATA5/$filename/ALT_results/noun$n.txt


#
sort EXP_DATA5/$filename/C/ALL$n.txt > EXP_DATA5/$filename/sort/ALL$n.txt
uniq -c EXP_DATA5/$filename/sort/ALL$n.txt > EXP_DATA5/$filename/uniq/ALL$n.txt
sort -r EXP_DATA5/$filename/uniq/ALL$n.txt > EXP_DATA5/$filename/results/ALL$n.txt
sort EXP_DATA5/$filename/C/NOUN$n.txt > EXP_DATA5/$filename/sort/NOUN$n.txt
uniq -c EXP_DATA5/$filename/sort/NOUN$n.txt > EXP_DATA5/$filename/uniq/NOUN$n.txt
sort -r EXP_DATA5/$filename/uniq/NOUN$n.txt > EXP_DATA5/$filename/results/NOUN$n.txt
```

```
echo "$n end"
done

echo "$filename end"
done
```

4.2 節、4.3 節、4.4 節の Naive Bayes による分類を行うシェルプログラムそれぞれの節で入力するファイルを変え実験した。

```
rm -r EXP_DATA4/N_BAYES
mkdir EXP_DATA4/N_BAYES
mkdir EXP_DATA4/N_BAYES/SCORE
mkdir EXP_DATA4/N_BAYES/SORT
for filename in ichiro koizumi watanabe
do
for n in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20
do
gcc naivebayes.c -lm
./a.out TRAINING_DATA2/ichiro/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/OTHER/ALL_RESULT.txt  \\
EXP_DATA4/$filename/results/TOP_ALL$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/ALL_$filename.txt
echo "TOP_$n.jpeg" >> EXP_DATA4/IMAGE/$filename/imgname.txt
./a.out TRAINING_DATA2/ichiro/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/OTHER/ALL_RESULT.txt  \\
EXP_DATA4/$filename/results/MID_ALL$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/ALL_$filename.txt
echo "MID_$n.jpeg" >> EXP_DATA4/IMAGE/$filename/imgname.txt
./a.out TRAINING_DATA2/ichiro/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/OTHER/ALL_RESULT.txt  \\
EXP_DATA4/$filename/results/LAST_ALL$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/ALL_$filename.txt
echo "LAST_$n.jpeg" >> EXP_DATA4/IMAGE/$filename/imgname.txt
#echo "$filename $n txt noun end"

./a.out TRAINING_DATA2/ichiro/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/NOUN_RESULT.txt \\
 TRAINING_DATA2/watanabe/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/OTHER/NOUN_RESULT.txt \\
EXP_DATA4/$filename/results/TOP_NOUN$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/NOUN_$filename.txt
./a.out TRAINING_DATA2/ichiro/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/NOUN_RESULT.txt \\
TRAINING_DATA2/watanabe/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/OTHER/NOUN_RESULT.txt \\
EXP_DATA4/$filename/results/MID_NOUN$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/NOUN_$filename.txt
./a.out TRAINING_DATA2/ichiro/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/NOUN_RESULT.txt \\
TRAINING_DATA2/watanabe/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/OTHER/NOUN_RESULT.txt \\
EXP_DATA4/$filename/results/LAST_NOUN$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/NOUN_$filename.txt

#echo "$filename $n  txt undef end"
./a.out TRAINING_DATA2/ichiro/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/ALL_RESULT.txt \\
TRAINING_DATA2/watanabe/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/OTHER/ALL_RESULT.txt \\
EXP_DATA4/$filename/ALT_results/TOP_all$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/ALTall_$filename.txt
./a.out TRAINING_DATA2/ichiro/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/OTHER/ALL_RESULT.txt  \\
EXP_DATA4/$filename/ALT_results/MID_all$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/ALTall_$filename.txt
```

```
./a.out TRAINING_DATA2/ichiro/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/OTHER/ALL_RESULT.txt  \\
EXP_DATA4/$filename/ALT_results/LAST_all$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/ALTall_$filename.txt

#echo "$filename $n ALT noun end"
./a.out TRAINING_DATA2/ichiro/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/OTHER/NOUN_RESULT.txt  \\
EXP_DATA4/$filename/ALT_results/TOP_noun$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/ALTnoun_$filename.txt
./a.out TRAINING_DATA2/ichiro/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/OTHER/NOUN_RESULT.txt  \\
EXP_DATA4/$filename/ALT_results/MID_noun$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/ALTnoun_$filename.txt
./a.out TRAINING_DATA2/ichiro/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/OTHER/NOUN_RESULT.txt  \\
EXP_DATA4/$filename/ALT_results/LAST_noun$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/ALTnoun_$filename.txt
#echo "$filename $n ALT undef end"

./a.out TRAINING_DATA2/ichiro/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/OTHER/ALL_RESULT.txt  \\
EXP_DATA4/$filename/results/TOP_all$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/all_$filename.txt
./a.out TRAINING_DATA2/ichiro/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/OTHER/ALL_RESULT.txt  \\
EXP_DATA4/$filename/results/MID_all$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/all_$filename.txt
./a.out TRAINING_DATA2/ichiro/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/koizumi/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/ALL_RESULT.txt  \\
TRAINING_DATA2/OTHER/ALL_RESULT.txt  \\
EXP_DATA4/$filename/results/LAST_all$n.txt  \\
EXP_DATA4/N_BAYES/SCORE/all_$filename.txt

./a.out TRAINING_DATA2/ichiro/RESULT/NOUN_RESULT.txt TRAINING_DATA2/koizumi/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/NOUN_RESULT.txt TRAINING_DATA2/OTHER/NOUN_RESULT.txt  \\
EXP_DATA4/$filename/results/TOP_noun$n.txt EXP_DATA4/N_BAYES/SCORE/noun_$filename.txt
./a.out TRAINING_DATA2/ichiro/RESULT/NOUN_RESULT.txt TRAINING_DATA2/koizumi/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/NOUN_RESULT.txt TRAINING_DATA2/OTHER/NOUN_RESULT.txt  \\
EXP_DATA4/$filename/results/MID_noun$n.txt EXP_DATA4/N_BAYES/SCORE/noun_$filename.txt
./a.out TRAINING_DATA2/ichiro/RESULT/NOUN_RESULT.txt TRAINING_DATA2/koizumi/RESULT/NOUN_RESULT.txt  \\
TRAINING_DATA2/watanabe/RESULT/NOUN_RESULT.txt TRAINING_DATA2/OTHER/NOUN_RESULT.txt  \\
EXP_DATA4/$filename/results/LAST_noun$n.txt EXP_DATA4/N_BAYES/SCORE/noun_$filename.txt

echo "$filename $n end"
done

gcc score_sum.c
./a.out EXP_DATA4/N_BAYES/SCORE/ALTnoun_$filename.txt  \\
EXP_DATA4/N_BAYES/SCORE/NOUN_$filename.txt  \\
EXP_DATA4/IMAGE/$filename/imgname.txt EXP_DATA4/N_BAYES/SCORE/$filename.txt

gcc sort.c
./a.out EXP_DATA4/N_BAYES/SCORE/noun_$filename.txt \\
```

```
EXP_DATA4/N_BAYES/SORT/sort_txtnoun_$filename.txt \\
./a.out EXP_DATA4/N_BAYES/SCORE/all_$filename.txt  \\
EXP_DATA4/N_BAYES/SORT/sort_txtall_$filename.txt
./a.out EXP_DATA4/N_BAYES/SCORE/ALTnoun_$filename.txt  \\
EXP_DATA4/N_BAYES/SORT/sort_ALTnoun_$filename.txt
./a.out EXP_DATA4/N_BAYES/SCORE/ALTall_$filename.txt  \\
EXP_DATA4/N_BAYES/SORT/sort_ALTall_$filename.txt
./a.out EXP_DATA4/N_BAYES/SCORE/ALL_$filename.txt  \\
EXP_DATA4/N_BAYES/SORT/sort_ALL_$filename.txt
./a.out EXP_DATA4/N_BAYES/SCORE/NOUN_$filename.txt  \\
EXP_DATA4/N_BAYES/SORT/sort_NOUN_$filename.txt
./a.out EXP_DATA4/N_BAYES/SCORE/$filename.txt  \\
EXP_DATA4/N_BAYES/SORT/$filename.txt

gcc make_html.c
./a.out EXP_DATA4/N_BAYES/SORT/$filename.txt EXP_DATA4/$filename.html
echo "$filename end"
done
```

HTML 文書から本文のみ抽出する C プログラム

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

   #define LINESIZE  15000

   void quit(char *);

   int main(int argc, char *argv[])
   {
     FILE *f1,*f2,*f3,*f4;
     char buf[LINESIZE], bun[LINESIZE],img[LINESIZE],alt[LINESIZE];
     int  r,x=0,y=0,n=0,m=0,i=0,j=0,k=0,flag1=0,flag2=0,flag3=0,flag4=0,flag5=0;

     if(argc != 5) quit("引数の数が違う  prog datafile ");
     if((f1 = fopen(argv[1],"r")) == NULL) quit("ファイルが開けない 1");
     if((f2 = fopen(argv[2],"w+")) == NULL) quit("ファイルが開けない 2");
     if((f3 = fopen(argv[3],"w+")) == NULL) quit("ファイルが開けない 3");
     if((f4 = fopen(argv[4],"w+")) == NULL) quit("ファイルが開けない 4");

     strcpy (buf,"");
     strcpy (bun,"");
         strcpy (img,"");
 strcpy (alt,"");

     while(fgets(buf,LINESIZE,f1) != NULL) {
  while (buf[i] != '\0') {
              if (buf[i]=='s' && buf[i+1]=='r' && buf[i+2]=='c' \\
                  && buf[i+3]=='=') {
            while (buf[i+5] != '.'  ){
          img[m]=buf[i+5];
          m++;i++;
    }
    img[m]= '\0';
    fprintf (f3,"%s\n",img);
    strcpy (img,"");m=0;
       }
              if (buf[i]=='a' && buf[i+1]=='l' && buf[i+2]=='t' && \\
                  buf[i+3]=='=') {
                             while (buf[i+5] != '"' ){
                             if (isspace(buf[i+5]) != 1 || \\
                                 iscntrl(buf[i+5]) != 1 \\
                                 || buf[i+5] != '\cdot'){
                        alt[y]=buf[i+5];
  }
                    y++;i++;
    }
    alt[y]= '\0';
    fprintf (f4,"%s\n",alt);
    strcpy (alt,"");y=0;
       }

       if (buf[i]== '}'){
    flag4=0;
       }
       if (buf[i]== '}'){
    flag4=0;
       }

       if (buf[i]== '{' || buf[0]=='.'){
    flag4=1;
       }
       if (buf[i]== '}'){
    flag4=0;
       }
```

```
      if (buf[i]== '<') {
   flag1=1;
   if(buf[i+1]=='s' && buf[i+2]=='c' && buf[i+3]=='r'
                     && buf[i+4]=='i' && buf[i+5]=='p' && buf[i+6]=='t'){
flag3=1;
   }
   if(buf[i+1]=='/' && buf[i+2]=='s' && buf[i+3]=='c'
                    && buf[i+4]=='r' && buf[i+5]=='i' \\
                    && buf[i+6]=='p' && buf[i+7]=='t'){
flag3=0;
   }
      }
      if (buf[i]=='>') {
   flag1=0;
      }

      if (flag1 == 0 && flag3==0 && flag4 ==0 && flag5==0
                    && buf[i]!= '<' && buf[i]!= '>' &&  \\
                     buf[i]!= '{' && buf[i]!= '}'
           && buf[i]!=':' &&buf[i]!='/' && buf[i]!='*' \\
           && buf[i]!='(' && buf[i]!=')'
           && strstr(buf,"var") == 0 && strstr(buf,"#") == 0 \\
           && isdigit(buf[i]) == 0) {
   if (buf[i]=='&'){
    flag2=1;
   }
                 if (buf[i]==';' && flag2==1 ){
flag2=0;
   }
   if (flag2==0 && buf[i]!= ';'){
                      bun[j]=buf[i];
            j++;
   }
      }
      i++;
 }
 bun[j]= '\0';
        if(strcmp(bun,"\0")!=0 ) {
             fprintf (f2,"%s\n",bun);
 }
 i=0;j=0;m=0;y=0;flag2=0;
 strcpy (bun,"");
 strcpy (buf,"");
   }
        if((r = fclose(f1)) == -1) quit("ファイルが閉じれない 1");
   if((r = fclose(f2)) == -1) quit("ファイルが閉じれない 2");
   if((r = fclose(f3)) == -1) quit("ファイルが閉じれない 3");
   if((r = fclose(f4)) == -1) quit("ファイルが閉じれない 4");
 }

 void quit(char *s)
 {
   puts(s); exit(1);
 }
```

juman による形態素解析結果から名詞未定義語のみ抽出する C プログラム

```
/* 名詞、未定義語のカウント */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define LINESIZE  500
#define WORDSIZE  100

void quit(char *);

int main(int argc, char *argv[])
{
  FILE *f1,*f2,*f3,*f4,*f5;
  char buf[LINESIZE], word1[WORDSIZE], word2[WORDSIZE];
  char word3[WORDSIZE],word4[WORDSIZE];
  int  r,n;

  if(argc != 6) quit("引数が違います prog datafile ");
  if((f1 = fopen(argv[1],"r")) == NULL) quit("ファイルが開けない 5");
  if((f2 = fopen(argv[2],"w")) == NULL) quit("ファイルが開けない 6");
  if((f3 = fopen(argv[3],"w")) == NULL) quit("ファイルが開けない 7");
  if((f4 = fopen(argv[4],"a+")) == NULL) quit("ファイルが開けない 8");
  if((f5 = fopen(argv[5],"a+")) == NULL) quit("ファイルが開けない 9");

  while (fgets(buf,LINESIZE,f1) != NULL){
        sscanf(buf,"%s %s %s %s",word1,word2,word3,word4);
 if (strcmp(word1,"EOS") != 0){
    fprintf (f2,"%s\n",word3);
            fprintf (f4,"%s\n",word3);
            if(strcmp(word4,"名詞") == 0) {
              fprintf(f3,"%s\n",word3);
      fprintf(f5,"%s\n",word3);
            }
            if(strcmp(word4,"未定義語") == 0) {
              fprintf(f3,"%s\n",word3);
      fprintf(f5,"%s\n",word3);
            }
 }
  }
  if((r = fclose(f4)) == -1) quit("ファイルが閉じれない 4");
  if((r = fclose(f3)) == -1) quit("ファイルが閉じれない 3");
  if((r = fclose(f2)) == -1) quit("ファイルが閉じれない 2");
  if((r = fclose(f1)) == -1) quit("ファイルが閉じれない 1");
}

void quit(char *s)
{
  puts(s); exit(1);
}
```

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<math.h>

#define LINESIZE 60
#define WORDSIZE 40
#define NUM 7000
#define CLASS 4
void quit (char *);

int main (int argc, char *argv[])
{
     FILE *f1,*f2,*f3,*f4,*fex,*fout;
     char buf[LINESIZE],word1[WORDSIZE];
     char x[NUM][WORDSIZE];
     int r,i=0,n=0,N=0,A=0,B=0,C=0,D=0,temp=0,c=0;
     double fxA[NUM],fxB[NUM],fxC[NUM],fxD[NUM];
     double cls[CLASS];
     double Pxc,SUM=0.0,SCORE=0.0;

  if(argc != 7) quit("引数が違います prog datafile ");
  if((f1 = fopen(argv[1],"r")) == NULL) quit("ファイルが開けない 1");
  if((f2 = fopen(argv[2],"r")) == NULL) quit("ファイルが開けない 2");
  if((f3 = fopen(argv[3],"r")) == NULL) quit("ファイルが開けない 3");
  if((f4 = fopen(argv[4],"r")) == NULL) quit("ファイルが開けない 4");
  if((fex = fopen(argv[5],"r")) == NULL) quit("ファイルが開けない 5");
  if((fout = fopen(argv[6],"a+")) == NULL) quit("ファイルが開けない 6");

  for (i=0; i<CLASS; i++){
       cls[i]=0.0;
  }i=0;
  strcpy (buf,"");
  while(i < NUM-1){
       strcpy (x[i],"");
       i++;
  }i=0;

  while (fgets(buf,LINESIZE,fex) != NULL){
       sscanf (buf,"%d %s",&temp,&x[i]);
       fxA[i]=1.0;
       fxB[i]=1.0;
       fxC[i]=1.0;
       fxD[i]=1.0;
       i++;
  }
  n=i-1;
  i=0;
  temp=0;

  while (fgets(buf,LINESIZE,f1) != NULL){
       sscanf (buf,"%d %s",&temp,&word1);
       for (i=0; i<=n; i++){
    if(strcmp(word1,x[i])==0){
    fxA[i]=(double)((fxA[i]-1.0)+temp);
    }
       }
       A=A+temp;
if(A < 0){
     printf ("word1=%s\n",word1);
     scanf("\n");
}
       N=N+temp;
       temp=0;i=0;
       strcpy(word1,"");
```

```
 }

 while (fgets(buf,LINESIZE,f2) != NULL){
      sscanf (buf,"%d %s",&temp,&word1);
      for (i=0; i<=n; i++){
          if(strcmp(word1,x[i])==0){
fxB[i]=(double)((fxB[i]-1.0)+temp);
i++;
   }
      }
      B=B+temp;
      N=N+temp;
      temp=0;i=0;
      strcpy(word1,"");
 }

 while (fgets(buf,LINESIZE,f3) != NULL){
      sscanf (buf,"%d %s",&temp,&word1);
      for (i=0; i<=n; i++){
   if(strcmp(word1,x[i])==0){
fxC[i]=(double)((fxC[i]-1.0)+temp);
        i++;
   }
      }
      C=C+temp;
      if (C<0){
   printf ("-");
   printf ("%d %s\n",temp,word1);
   scanf ("%d",i);
      }
      N=N+temp;
      temp=0;i=0;
      strcpy(word1,"");
 }

 while (fgets(buf,LINESIZE,f4) != NULL){
      sscanf (buf,"%d %s",&temp,&word1);
      for (i=0; i<=n; i++){
   if(strcmp(word1,x[i])==0){
fxD[i]=(double)((fxD[i]-1.0)+temp);
        i++;
   }
      }
      D=D+temp;
      N=N+temp;
      temp=0;i=0;
      strcpy(word1,"");
 }
 Pxc=0.0;
 for (i=0; i<=n; i++){
      Pxc=fabs(log((double)(fxA[i]/A)))+Pxc;
 }
 cls[c]=((double)A/N)*Pxc;
 c++;
 Pxc=0.0;i=0;

 for (i=0; i<=n; i++){
      Pxc=fabs(log((double)(fxB[i]/B)))+Pxc;
 }
 cls[c]=((double)B/N)*Pxc;
 c++;
 Pxc=0.0;i=0;

 for (i=0; i<=n; i++){
      Pxc=fabs(log((double)(fxC[i]/C)))+Pxc;
 }
```

```
  cls[c]=((double)C/N)*Pxc;
  c++;
  Pxc=0.0;i=0;


  for (i=0; i<=n; i++){
       Pxc=fabs(log((double)(fxD[i]/D)))+Pxc;
  }
  cls[c]=((double)D/N)*Pxc;
  c++;
  Pxc=0.0;i=0;

  for(i=0;i<CLASS; i++){
       SUM=cls[i]+SUM;
  }
  if(strstr(argv[5],"ichiro") !=0 && SUM !=0.0){
       SCORE=cls[0]/SUM;
  }
  if(strstr(argv[5],"koizumi") !=0 && SUM !=0.0){
       SCORE=cls[0]/SUM;
  }
  if(strstr(argv[5],"watanabe") !=0 && SUM !=0.0){
       SCORE=cls[0]/SUM;
  }


  fprintf (fout,"%s ",argv[5]);
  for(i=0; i<CLASS; i++){
       fprintf (fout," %f ",cls[i]);
  }
  fprintf(fout,"\n");

}

void quit (char *s)
{
     puts(s);exit(1);
}
```