Vol.2014-NL-215 No.8
2014/2/6

難易度及び類似度を用いた コンピューター関連書籍推薦システムの開発

舟木類佳[†1]　　黒田久泰[†1]

コンピューター関連書籍を学習する者にとって書籍がやさしいか，難しいかといった指標は重要である．そこで本研究では難易度による書籍の推薦システムの開発を行った．難易度の比較は類似度が高い書籍間で行われると考え，本研究ではまず，潜在的ディリクレ配分法及びコサイン類似度によって目次及び書名の類似度を求めた．その後，予め難易度付与を行った一部の書籍に対して特徴量を定義し Support Vector Machine により難易度を予測した．この予測値を用いて難易度を推測し，類似度が高い書籍同士の難易度比較を行った．そして，Ruby on Rails を用いて書籍の検索アプリケーションを開発した．

Development of IT Books Recommendation System Using Difficulty and Similarity

RUKA FUNAKI[†1]　　HISAYASU KURODA[†1]

Indicator whether a book is difficult or easy is important for the people who learn IT books. Therefore, in this research, we developed book recommendation system by book difficulty. First, considering difficulty is compared between similar books and one another, we evaluated similarity of list of contents and book title by Latent Dirichlet Allocation and cosine similarity. Afterward, we evaluated predicated value by Support Vector Machine with some books we added difficulty in advance. With this predicated value, we calculated difficulty and compared difficulty in similar books with one another. In addition, we developed book search application with Ruby on Rails.

1. はじめに

近年情報処理の技術の進歩はめざましく，多種多様の技術を含むようになってきた．それに伴い，様々な分野の様々な技術に関する書籍が出版され，複雑さを増している．また，Amazon.com[a)]や楽天ブックス[b)]の利用をはじめとする Web サイト上で書籍を購入する機会が増えている一方で，学習者がコンピューター関連技術を学ぶ際に適切な書籍を選択することは困難である．

推薦システムで重要なのは，ユーザーの目的に沿った書籍を推薦できるかということである．ここでフィルタリングに用いられる書籍の属性として挙げられるのは分野，価格，出版社，難易度，著者など様々であるが，このうち分野と難易度によるフィルタリングは難しい．書籍の分野はある程度カテゴリ分けされていることが多いが，情報処理技術は分野が多岐に渡るため，書籍はより細分化された分類が求められる．また，難易度は著者や出版社によりランク付けされていないため，難易度が低い本のみを抽出することは難しい．このことから，本研究では難易度及び書籍の類似度に着目した検索システムを構築した．

先に上げた Amazon.com では「この商品を買った人はこんな商品も買っています」といった文言で推薦を行うシステムを備えている．本研究では「この本よりやさしい本ではこれがおすすめです」，「この分野を更に深めたいならこの本がおすすめです」などの難易度を考慮した推薦手法を提案する．また，本研究では目次と書名を手がかりに類似度及び難易度を推定する．

2. 研究手順

2.1 研究の概要

難易度を比較する場合に，全く異なる分野同士の書籍を比較することに意味はない．つまり，難易度を比較するには書籍が同分野であると考えた．この仮定に従い，書籍の(1)類似度を計算した後に，(2)難易度を計算し，(3)学習者に適した書籍を推薦する手法を提案する（図 1）．システムの開発は R 言語を用いて行った．

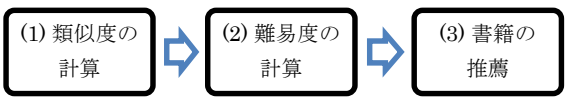

図 1　研究手順

ところで書籍の内容はすべて公開されているわけではなく，電子化もいまだ進んでいない．そのため，類似度判定及び難易度判定の手がかりは公開されている目次と書名のみとなる．本研究では予め人手によって難易度付与を行ったデータに対して機械学習を行うことで，人間が行う「難しそう」「簡単そう」といった難易度推定をどれだけ模倣できるかを検証する．

†1 愛媛大学
Ehime University
a) http://amazon.com/
b) http://books.rakuten.co.jp/

2.2 研究の流れ

本研究の流れは以下に示す通りである．

(1) Web スクレイピングにより書籍の書名，目次情報を取得する
(2) 書籍の書名と目次情報に対して形態素解析を施し，ストップワードを除去する
(3) 形態素解析結果から単語-文書行列を作成する
(4) LDA により文書を複数のトピックとして表現する
(5) コサイン類似度による類似度を計算し，類似度の高いものを同じ分野の書籍として推定する
(6) 品詞や文字種，及び単語による特徴量抽出を行う
(7) SVM によって学習し，それにより難易度判定を行う
(8) 類似度及び難易度を用いた推薦を行う

3. 関連技術

3.1 目次を用いた研究

書籍の目次をクラスタリングに用いた研究は少ないが，目次を使った研究の例として，石田ら[1]の研究がある．彼らは書名と目次，帯の情報を用いて相対出現率と相互情報量に基づく重み付けと SVM による図書の日本十進分類法（NDC）への分類を行っている．彼らはこの中で，目次と帯の情報が書名を補間する役割があることを示している．

3.2 難易度判定に関する研究

英語の文章を対象として難易度を測る手法は古くから研究がされており，Flesch Kincaid Grade Level[2]や Dale-Chall 法[3]などがある．Flesch Kincaid Grade Level は文の長さと単語の音節数の統計に基づき，小学 1 年生を 1，中学 1 年生を 7 としたスコア付けを行っている．また Dale-Chall 法は基本単語のリストに含まれない難語の割合と文の長さに基づきスコア付けを行っている．これらはこうした特徴量をもとに回帰手法を用いてスコアを算出する方法である．

また，Chorins-Thompson ら[4]は統計モデルを用いており，Schwarm[5]は SVM を用いている．また，田中ら[6]は SVM を用いた難易度比較器を構築し，難易度順にソートする手法を提案した．また，日本語の文章においてもリーダビリティーの研究が行われており，佐藤ら[7]は教科書コーパスを文字の出現頻度，柴崎ら[8]は文章中の平均文字数，平均文節数，品詞の割合，語義数などを用いている．また MetaMetrics 社による Lexile 指数[c]という難易度指標も存在する．しかし，これら多くが一般書籍を対象としており，コンピューター関連書籍の難易度に適応できるかどうかは定かではない．なぜならコンピューター関連書籍は分野自体の難易度が高く，専門性が高いものだからである．そこで，本研究では柴崎らの手法に習い，品詞，文字種を特徴量とした場合を考え，さらに単語ベクトルを特徴量とした場合を考える．また，学年を推定するものとは異なり指標が存在しないため，書籍の難易度を 4 段階に分けて考えた．

c) http://www.lexile.com/

4. 類似度の計算

4.1 書名および目次データの取得

Web 上に公開されている書名及び目次データを Web スクレイピングによって抽出，収集するプログラムを Ruby 言語で開発し，10,598 冊のコンピューター関連書籍の書名及び目次を収集した．

4.2 形態素解析による用語抽出

英文は単語区切りとしてスペースがあるため，スペースごとに区切るのみで単語を切り出すことができる．しかしながら日本語においては単語の境界は曖昧である．そのため，単語ごとに切り出す形態素解析が必要である．形態素解析ツールには MeCab[d]や茶筌[e]などが公開されているが，本研究では MeCab を用いた．

また，今回の類似度の計算は内容の類似度であるため，文書の表現的要素は必要ない．そこで，動詞や形容詞，助詞などの単語は文書の類似度計算に有益ではないと考え，名詞のみを抽出することとした．また，ストップワードとなりうる語（例えば，「前書き」や「章」「節」といった単語）を可能な限り取り除いた．

文書中の単語を特徴として bag-of-words 表現に基づく単語-文書行列を用いた．また，LDA に学習させるのに適したデータ形式に変換した．

4.3 崩壊型ギブスサンプリングを用いた LDA

行列の次元を削減する次元縮約の方法として，特異値分解により行列の次元を縮約する潜在的意味インデクシング法（LSI: Latent Semantic Indexing）や，T. Hofmann によるトピックモデルを用いた確率的潜在意味インデクシング法（PLSI: Probabilistic Latent Semantic Indexing）[9]及び，D. M. Bleiらの潜在的ディリクレ配分法（LDA: Latent Dirichlet Allocation）[10]がある．これらにより同じ意味，または近い意味で表記が異なる単語をまとめることができる．

LDA は Blei が 2003 年に発表したアルゴリズムであり，文書が，複数のトピック分布によって構成されるというものである．Blei は変分ベイズ法による LDA モデルの推定を行っている．そののち 2004 年に崩壊型ギブスサンプリング[11]による手法が提案された．

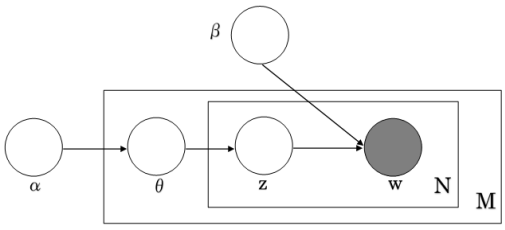

図 2　LDA のグラフィカルモデル

d) http://mecab.sourceforge.jp/
e) http://chasen-legacy.sourceforge.jp/

図2のようなグラフィカルモデルに従ってパラメータを推定し，トピックを抽出する．LDAではパラメータ α, β により文書は以下の課程で生成されると仮定する．

1. 単語数Nをポアソン分布に従い選択する

$$N\sim\text{Poisson}(\xi)$$

2. トピック分布θをDirechlet分布に基づき選択する

$$\theta\sim\text{Dir}(\alpha)$$

3. N個のそれぞれの単語について以下を繰り返す

a) トピックz_nを多項分布に基づき選択する

$$z_n\sim\text{Multinomial}(\theta)$$

b) トピックz_nに対する単語の確率分布に従って単語w_nを選択する

$$w_n\sim\text{P}(w_n|z_n,\beta)$$

以上の課程をM回繰り返して文書を生成する．α, βは崩壊型ギブスサンプリングを用いて学習する．初期値は $\alpha = 0.1$ および $\beta = 0.1$ とした．また，トピック数 $K = 100$ として学習を行った．これにより，文書がトピックを含む確率を求めることができる．図3はアルゴリズムに関する書籍である「アルゴリズムイントロダクション第１巻」をLDAによって分析し，書籍のトピック割合を円グラフにしたものである．ラベルにはトピックの代表的な単語が示されており，アルゴリズムに関連する単語が並んでいる．

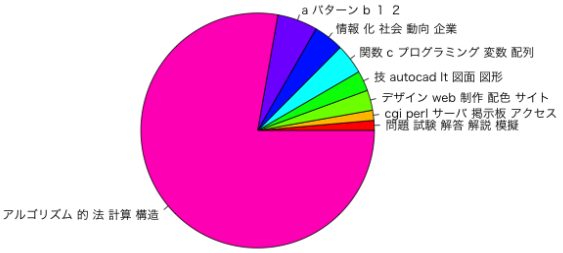

図 3　書籍「アルゴリズムイントロダクション第１巻」のトピック割合

4.4 コサイン類似度による類似度計算

異なる 2 つの文書同士の類似度を測る手法としてよく知られているものにコサイン類似度がある．長さの同じ特徴ベクトル $x = (x_1, x_2, \cdots x_i, \cdots, x_n)$，$y = (y_1, y_2, \cdots y_i, \cdots, y_n)$ があるとき，コサイン類似度は以下のように表される．

$$s_{cos}(x,y) = \frac{\sum x_i y_i}{\sqrt{\sum {x_i}^2 \sum {y_i}^2}}$$

LDA によって次元削減されたデータをコサイン類似度によって計算し，書籍の類似度を表す相関行列を生成した．また，その類似度を基に，書籍を類似している順に出力する関数を構築した．

5. 難易度の計算

5.1 品詞と文字種に対する仮定

先に述べたように，書籍の難易度を直接文章から測ることはできないため今回は書名および目次を用いて難易度を推定することとする．ここで，書名および目次は書籍に対する難易度情報を含むと仮定した．書名は「１週間でできる○○○」などといった書名から，難易度は比較的簡単であると読み取れる．また，目次に「やってみよう！」などといった砕けた表現を含む場合，簡単そうであると読み取れる．そうした情報をもとに，人間は書名や目次から難易度を推定していると考えた．こうした人間が書籍に対して行う難易度の推定を以下のような 5 つの仮定に分割した．

仮定1. 書名および目次に平仮名が多く含まれる場合，その書籍は難しい漢字や外来語を使っていないため難易度は低い

仮定2. 書名及び目次に片仮名や漢字，英字が多く含まれる場合，専門用語が多く含まれていると考えられるため，その書籍の難易度は高い

仮定3. 「！」などの記号を多く含む場合に砕けた表現を多く含むと考えられるため，その書籍は難易度が低い

仮定4. 書籍及び目次に名詞が多く含まれる場合，専門用語が多く含まれており，固い表現が多いと考えられるためその書籍の難易度は高い

仮定5. 動詞や形容詞，副詞，感動詞を多く含む場合，砕けた表現を多く含むと考えられるためその書籍の難易度は低い

まず，10,598 冊のコンピューター関連書籍のうち，1,000 冊に対して予め難易度を付与した．我々が書名，目次，書籍の表紙画像のみを情報とし，「入門レベル」「初心者レベル」「中級者レベル」「上級者レベル」の 4 段階で難易度付与を行った．ここで，分類した難易度の分布は表 1 のようになった．

表 1　難易度の分布

難易度	レベル	分布
1	入門レベル	189
2	初心者レベル	256
3	中級者レベル	389
4	上級者レベル	166

5.2 スコアと難易度の順位相関

5.1 節の仮定に基づき，文字種によるスコア付け，及び文の品詞によるスコア付けをした．まず，文字種によるスコア付けとは平仮名，片仮名，漢字，英字，数字，記号の全体に対する割合をスコアとしたものである．また，品詞によるスコア付けとは名詞，動詞，形容詞，副詞，助詞，接続詞，助動詞，連体詞，感動詞による全体の単語に対する割合をスコアとしたものである．

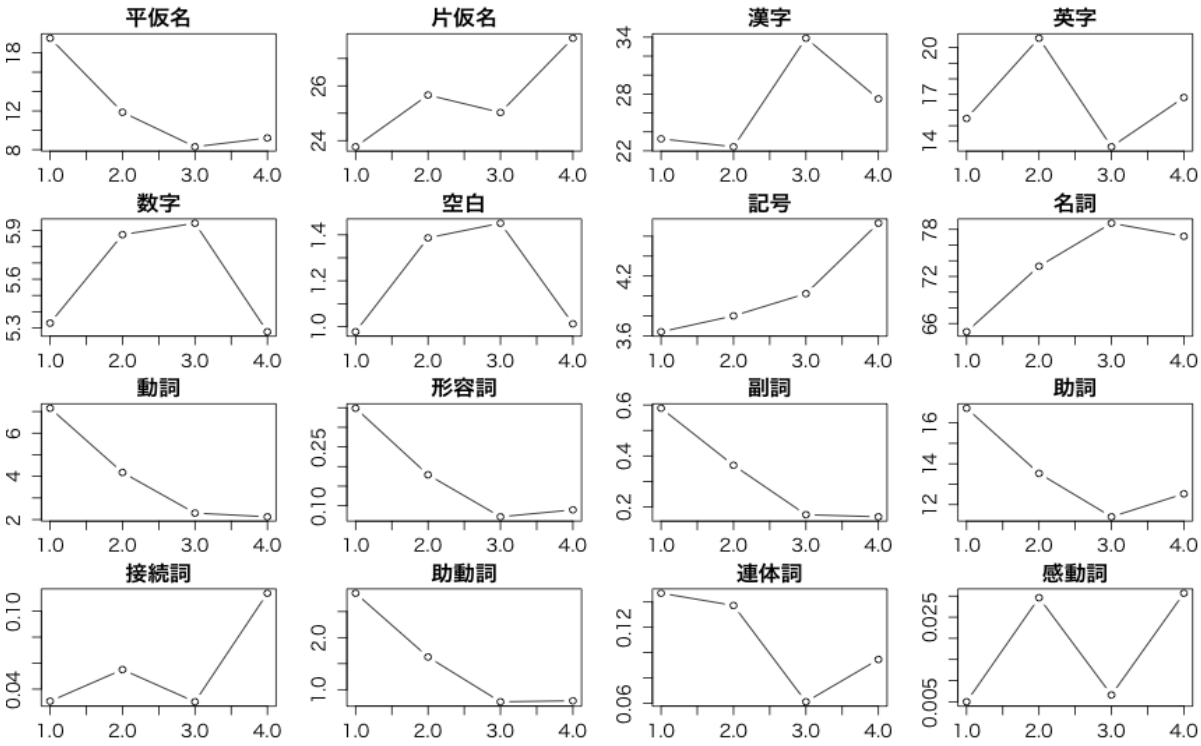

図 4 文字種と品詞のスコアと難易度の関係（縦軸は構成要素の割合（%），横軸は難易度（4 段階）を示す）

表 2　スコアと難易度の順位相関係数

構成	相関係数 r	P 値	|r|>0.1	P 値<0.05
平仮名	-0.3934	< 2.2e-16	○	○
片仮名	0.0530	0.0932	×	×
漢字	0.1633	2.038e-07	○	○
英字	-0.0871	0.0057	×	○
数字	0.0046	0.8842	×	×
空白	0.0429	0.1749	×	×
記号	0.1292	4.121e-05	○	○
名詞	0.3651	< 2.2e-16	○	○
動詞	-0.4116	< 2.2e-16	○	○
形容詞	-0.1711	5.119e-08	○	○
副詞	-0.2311	1.305e-13	○	○
助詞	-0.2614	2.2e-16	○	○
接続詞	0.0090	0.7746	×	×
助動詞	-0.3251	< 2.2e-16	○	○
連体詞	-0.0819	0.0094	×	○
感動詞	0.0326	0.3019	×	×

表 2 はそれぞれのスコアと難易度のスピアマンの順位相関係数rを表す．P 値が有意水準 5%より低い場合には相関係数は有意でないと判断する．また，相関係数rの絶対値が 0.1 以下の場合相関がないとみなす．図 4 はそれぞれ要素の全スコアの平均（縦軸）と難易度（横軸）のグラフを表している．表 2 及び図 4 から分かるように平仮名，漢字，記号に相関がある．そして平仮名や記号が多いほど難易度が低い傾向があり，漢字が多いほど難易度が高い傾向にある．これは仮定 1，仮定 2 及び仮定 3 と一致する．ただし，片仮名は図で見ると相関があるように見えるものの，有意と判断されなかった．

また，名詞，動詞，形容詞，副詞，助詞，助動詞に相関がある．そして名詞が多いほど難易度が高く，動詞，形容詞，副詞，助詞，助動詞が多いほど難易度が高くなる傾向にあることが見て取れる．これらは仮定 4 及び仮定 5 に一致する．ただし，感動詞に関する仮定は外れている．感動詞は書名や目次に含まれること自体まれであり，必要なサンプル数が得られなかったため有意な結果とならなかったと想定される．また，予め想定していなかった助詞や助動詞が難易度に影響することが分かった．これらの品詞が負の相関を持っているのは，砕けた表現において多く用いられるためであると考えられる．以上の相関の傾向からこれらのスコアを用いることで難易度を推定できると判断した．

文の文字種によるスコア，および文の品詞によるスコアを合わせて$x = (x_1, x_2, \cdots x_i, \cdots, x_n)$というベクトルで表し，特徴量とした．

5.3 単語ベクトルによる特徴量

書籍の単語には「やさしい」「プロフェッショナル」など難易度と密接に関係している単語は多いと思われる．そのため単語の出現頻度を特徴量とすることは意味があると考えた．形態素解析を施したすべての単語の tf-idf 値を用いてベクトルを作成し，それらを特徴量として用いた．個々で tf-idf 値は以下のように表される．tf(term frequency)はテキストにおける語の頻度，df(document frequency)は語を含むテキストの数，Nはテキストの総数を表す．

$$\text{tf-idf} = tf \times \log(\frac{N}{df})$$

5.4 SVM による学習

Support Vector Machine（SVM）とは 2 クラスのパターン識別手法であり，学習データから各データの距離（マージン）を最大化する超平面を求めることで高い識別性能を得ることができる．また，多値分類にも利用されている．

SVM を用いて難易度を学習するために予め付与した難易度をクラスとした書籍データ 1,000 件のうち，500 件を学習用データとし，残りの 500 件を評価用とした．ライブラリは LibSVM[12]を用いてカーネル関数には線形カーネル関数を用いた．

品詞及び文字種による特徴を用いた結果を表 3 に示す．行が正解の難易度，列が分類された難易度を示す．正解率は $(49 + 43 + 138 + 6) \div 500 = 47.2\,\%$ となった．

表 3　SVM の結果（品詞と文字種によるもの）

正解＼予測	1	2	3	4
1	49	22	14	4
2	22	43	37	20
3	25	50	138	60
4	0	8	2	6

単語ベクトルを用いた結果を表 4 に示す．正解率は $(59 + 84 + 150 + 47) \div 500 = 68.0\,\%$ となった．また，tf-idf 値ではなく tf 値を用いた場合には62.8%となり，tf-idf を用いたほうが良い結果が得られた．

表 4　SVM の結果（単語ベクトルによるもの）

正解＼予測	1	2	3	4
1	59	7	2	3
2	28	84	35	13
3	6	29	150	21
4	1	5	10	47

これらにより文字種や品詞をスコアとしたものは表 3 において対角線上に数値が集中していることからある程度正しく推定できていると分かる．しかし，単語ベクトルを特徴量として用いたほうが分類に効果的であることが分かった．1,000 冊の単語ベクトルを SVM により学習し，難易度が付与されていない残りの 9,598 冊の書籍の難易度を判定した．

5.5 類似度と難易度を用いた推薦

まず，LDA 及びコサイン類似度により類似度が高い書籍同士を得ることができるようになった．また，すべての書籍に難易度が付与された．これにより類似度が近く，難易度が目的の難易度に一致したものを推薦することができるようになった．

6. Web アプリケーションとしての実装

6.1 Ruby on Rails を用いた書籍検索システム

Ruby on Rails[f]を用いて書籍の推薦サイトを構築した．Ruby on Rails とは Ruby 言語によるオープンソースの Web アプリケーションフレームワークであり，現在多くの Web アプリケーションが Ruby on Rails を用いて開発されている．

RinRuby[g]は R 言語を Ruby 言語から扱うことのできるライブラリの一つである．このライブラリを用いることにより類似度及び難易度の計算に使った R 言語のプログラムを呼び出すことが可能である．しかしながら，毎回類似度の計算のためにすべてのプログラムを呼び出すことが非常に大きなオーバーヘッドとなり，Web アプリケーションとして実用的とは言えなくなってしまう．そのため，類似している書籍のランキングデータや難易度の推定データは予め計算してデータベースに格納し，R 言語を呼び出すことは最小限にするようにした．

6.2 操作画面

図 5　書籍一覧・検索画面

図 5 は書籍の検索画面である．テキストボックスに検索用語を入力することによって書名と目次からキーワード検索を行うことができるようになっている．また，書名や書籍の難易度で並び替えることも可能である．図では「ruby」というキーワードを書名に含む書籍を検索している．

書籍を検索し書籍名をクリックすることで，書籍の詳細画面に遷移する．図 6 は「Ruby on Rails アプリケーションプログラミング」という書籍の詳細画面である．書名と目次が表示されている．また，Web 上から書籍の表紙画像を取得，表示している．下の方にはこの書籍がどのようなトピックを含むのかの円グラフを生成し，表示している．

図 7 は図 6 に続く画面であり書籍の詳細画面の下に表示

f) http://rubyonrails.org/
g) http://rinruby.ddahl.org/

されている．ここでは該当の書籍に対して類似度の高い書籍を 30 件類似度順に上から並べて推薦している．図では「Ruby on Rails3 アプリケーションプログラミング」に類似する Ruby に関連する書籍が並んでいることが確認でき，類似度による推薦が正しく実装がされていると判断できる．

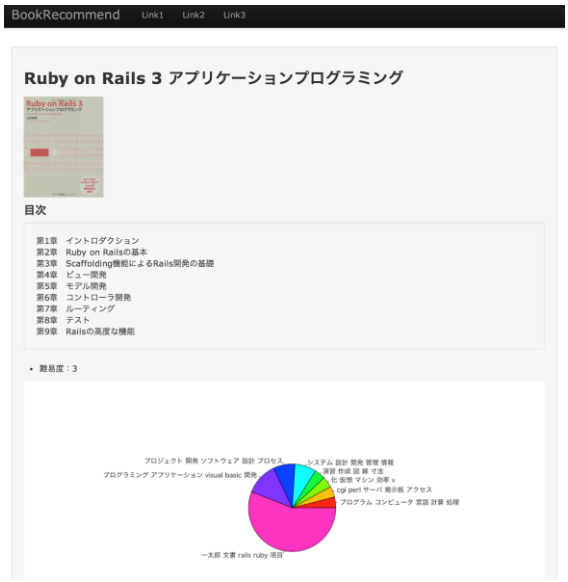

図 6　詳細画面

図 7　類似書籍の推薦画面

図 8　難易度を指定した推薦画面

図 8 は書籍の難易度が 1 であるもののみを類似順に表示したものである．一番上に表示されている「Heroku ではじめる Rails プログラミング入門」は Ruby on Rails の入門書であり，目次には「作ってみよう」などの砕けた表現が見られるため，簡単な書籍であると判断できる．それ以外も入門的な書籍が多く並んでいることが確認でき，難易度推薦が実装されていることが確認できる．

7. おわりに

類似度及び難易度による書籍の推薦システムを開発した．これにより，より適切な難易度の書籍をユーザーが見つけることができるようになった．今後は SVM の結果から重みベクトルを求め，影響力の大きい単語を見つける．また，難易度を自分で選ぶのではなく，自動で難易度を選択するなどより良いインタフェースを目指す．更に，LDA によって得られたトピックを基にユーザーに興味がありそうな単語を推薦し，ユーザーが単語を順次クリックしていくことで，選んだ単語が所属するトピックを多く含む目的の書籍を見つけることのできる機能なども考えられる．

参考文献

1) 石田栄美，宮田洋輔，神門典子，上田修一: 目次と帯を用いた図書の自動分類，情報処理学会情報学基礎研究会報告, Vol.2006, No.33, pp.85-92, 2006.
2) J.P.Kincaid, R.P. Fishburne, R. L. Rogers, and B.S. Chissom: *Derivation of new readability formulas for Navy enlisted personnel.* Research Branch Report 8-75, U. S. Naval Air Station, 1975.
3) J. S. Chall and E. Dale: *Readability revisited: The new Dale-Chall readability formula*. Brookline Books, 1995.
4) K. Collins-Thompson and J. Callan: *A language modeling approach to predicting reading difficulty.* Proceedings of HLT-NAACL, pp.193-200, 2004.
5) Sarah E. Schwarm, and Mari Ostendorf: *Reading level assessment using support vector machines and statistical language models*, Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pp.523-530, 2005.
6) Satoshi Sato, Matsuyoshi Suguru. and Kondoh Yohsuke: *Automatic Assessment of Japanese Text Readability Based on a Textbook Corpus*, LREC, 2008.
7) 柴崎秀子，沢井康孝: 国語教科書コーパスを応用した日本語リーダビリティー構築のための基礎研究，信学技報 NLC2007-32, , Vol.107, No.246, pp.19-24, 2007.
8) Kumiko Tanaka-Ishii, Satoshi Tezuka , and Hiroshi Terada: *Sorting texts by readability*, Computational Linguistics, Vol.36, No.2, pp.203-227, 2010
9) Thomas, H.: *Probabilistic Latent Semantic Indexing*, Proceedings of the 22nd Annual International SIGIR Conference on Research and Development in Information Retrieval, 1999.
10) David M. Blei, Andrew Y. Ng, and Michael I. Jordan: *Latent Dirichlet allocation*. In Lafferty, John. Journal of Machine Learning Research 3 (4–5): pp.993–1022, 2003.
11) T.L. Griffiths and M. Steyvers：Finding scientific topics. Proceedings of the National Academy of Sciences of the United States of America, Vol. 101, No. Suppl 1, pp. 5228–5235, 2004.
12) Chih-Chung Chang, and Chih-Jen Lin: *LIBSVM: A library for support vector machines*, ACM Transactions on Intelligent Systems and Technology (TIST), Vol.2, No.3, pp.1-27, 2011.