Vol.2012-NL-206 No.13
Vol.2012-SLP-91 No.13
2012/5/11

リアルタイムクラウドソーシングと Twitter大規模コーパスを利用した対話システム

別所 史浩[1,a)]　原田 達也[1,b)]　國吉 康夫[1,c)]

概要：Twitter 大規模コーパスとリアルタイムクラウドソーシングの枠組みを利用した対話システムを提案する．我々のシステムは複雑な対話管理を行わず，ユーザ発話に対して最も近い発話をデータベース中から探し出し，それに対する応答をシステム応答として返す枠組みとなっている．またデータベース内に適切な応答が見つからなかったときには，その発言を即時に他ユーザにクラウドソースする．その際ユーザが匿名化され，かつクラウドソースをしていることを認知させないような仕組みを提案する．本論文ではまず，Twitter から日本語発話対を抽出し，内容を解析する．得られたデータの中で長さが 2 を超えるものは発話対全体の 58.3%であった．得られた発話対からの類似発話検索では，各種応答選択手法に対して AUC (Area Under the ROC Curve) を用いた性能の比較評価を行い，品詞フィルタリング，文書正規化，本研究における学習データの分類性能の向上に有効に働くことが示された．実装されたシステムの評価実験を通じて，システムの用いるデータ数の増加は，多くの場合システム性能の向上に寄与するが，一定の値において有意に性能が下落することが観測された．また，クラウドソースを含めた実験を通じて，クラウドソースの枠組みの導入でユーザがシステムとの対話の中に面白さを感じるようになることが示された．

キーワード：自然言語処理，チャットボット，類似文書検索，文書ベクトル，ROC 曲線，AUC

Dialog System Using Real-Time Crowdsourcing and Two-Length Tweet Corpus

Bessho Fumihiro[1,a)]　Harada Tatsuya[1,b)]　Kuniyoshi Yasuo[1,c)]

Abstract: We propose a dialog system that creates responses based on a 2-length tweets database and real-time crowdsourcing. Our system replies with the utterance from the database that is most similar to the user input. We also propose a real-time crowdsourcing framework for handling the case in which there is no adequate response in the database. The response scoring function is designed and evaluated using a survey, based on which positive/negative utterance pairs are created. We examine the effect of data size and real-time crowdsourcing on system response. Our results show that system performance improves with increasing amount of data in many case, but not always, and crowdsourcing framework enhances amusingness of the system.

Keywords: Natural Language Processing, Chat Bot, Similar Document Retrieval, Document Vector, ROC curve, AUC

1. はじめに

近年，家庭内へのコミュニケーションロボットの浸透や，昨今の各種情報端末の普及によってロボットの一要素として，また，エンターテインメントのための一つのアプ

[1] 東京大学
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

a) bessho@isi.imi.i.u-tokyo.ac.jp
b) harada@isi.imi.i.u-tokyo.ac.jp
c) kuniyosh@isi.imi.i.u-tokyo.ac.jp

Vol.2012-NL-206 No.13
Vol.2012-SLP-91 No.13
2012/5/11

リケーションとして，対話システムの需要が高まっている．一方でインターネット上には言語データが溢れている．例えば Twitter は投稿された情報を収集・検索するための API を多く提供しており，多くの研究でその言語データが用いられている．例えば，株式市場予測 [2] や，ソーシャルメディアにおける情報の広がり [1]，品詞タグ付け [4]，意見抽出 [6] などが行われている．また，データサイズの大きさと，各投稿に対してユーザが返信 (reply) できるため，発言が対話形式になりやすいという性質から，対話モデル [5], [10] の構築も行われている．

[7] に示されるように，英語のツイートのうち 37%は対話形式 (Conversationl) である．また，そのうち 69%は 2 つのツイート，すなわちある発言とそれに対する応答からなる．我々の行った 250 万を超えるツイートの解析では，英語同様日本語ツイートの 37.5%は対話形式である．ただし，2 つのツイートからなるものはそのうち 58.3%に満たないという結果となった．

本論文で提案する対話システムは特定のタスクの遂行を目的としないいわゆるチャットボット (chat bot) である．A.L.I.C.E[11] などこれまでの多くのチャットボットはルールベースのものが主流であったが，近年では大規模コーパスに基づくチャットボット[*1][*2]も提案されている．ただし，これらのチャットボットはルールに記述されていなかったりデータベース中にデータのない種類のユーザ発話に対しては適切に対応することが出来ないという問題があった．

本論文ではリアルタイムクラウドソーシングとツイッター大規模データを用いた対話システム（チャットボット）を提案する．リアルタイムクラウドソーシングを利用することで，データベースにない種類のユーザ発話に対しても適切な応答を得ることができ，これが本研究の第 1 の貢献である．加えて，Twitter 中の日本語で書かれた対話形式データについて，全データに対する割合や発話長分布の解析を行ったことも本研究の貢献である．

2. 方法

2.1 概要

我々は「発話対」データベースを図 **1** に示されるように設計した．各ペアはある発話と，それに対する応答からなる．システムが応答を生成する方法は単純であり，図 **2** のように表される．それぞれのユーザ発話に対して，システムは発話対データベースの中で，最も近い発話（図 1 中の発話 A）を検索する．この発話に対応する応答（図 1 中の発話 B）がシステムの応答として選択される．システムがユーザの発話と十分に近い発話をデータベース中に発見できなかった場合，システムはユーザの発話を他のユーザへ「アウトソース」する．

[*1] Jabberwacky: `http://www.jabberwacky.com`
[*2] Cleverbot: `http://www.cleverbot.com`

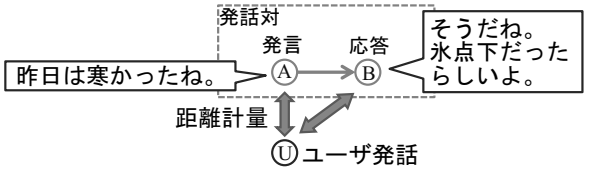


図 **1** 発話対
Fig. 1 Utterance pair.

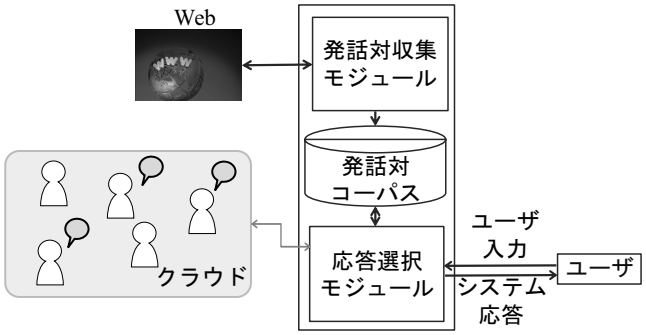


図 **2** システム概要
Fig. 2 System overview.

チャット風システム	
秒	人数
2-10	55
30-45	20
90	25

ツイッター風システム	
秒	人数
10	5
60	15
90	34
120-300	34
1200-3600	10

表 **1** ユーザが許容するシステム応答時間
Table 1 Response time allowed by users.

対話システムにおいて，応答速度は重要な要素である．我々は本システムについて 2 種類の使用方法を想定し，それぞれについて適切な応答時間を検証した．2 種類の使用方法とはすなわち，チャット風システム，ツイッター風システムの 2 つである．チャット風システムとは，従来のチャットと同様に，システムとの対話中においてユーザが常にプロンプトと向きあうようなシステムである．一方でツイッター風システムとは，ツイッターやメールのように，他の仕事の片手間としてユーザが対話を行うようなシステムである．この 2 種類のシステムの使用方法において，ユーザが異なるシステムの適切な応答時間を想定すると考えるのは自然である．実際にこれらの応答時間についてアンケートを行った．アンケートは 4.2 節で述べる統合実験の実験後，被験者に対して行ったアンケートの中で行ったものである．回答人数は延べ 100 名であるが，無回答があったため，総数は合わない．アンケートは結果を表 **1** に示す．過半数 (55%) の被験者はチャット風システムにおいて 2-10 秒の応答を適切な応答時間とし，84%の被験者はツイッター風システムにおいて 60-300 秒を適切な応答時間と回答した（表 1）．

2.2 データ

対話データベースとして，我々は 120 万の発話対を Twitter の API を利用して収集した．我々はパブリックタイムラインのデータを Streaming API [*3] を利用することで，また，それらのデータのうち日本語で書かれており[*4]かつ，in-reply-to 欄が存在する，すなわち他の発言の返信となっている発言について，その応答元となる発言を REST API [*5]. を用いて収集した．

Twitter 中の発言は，メンション（返答先のユーザ名を示すために用いられ，アットマークとそれに続くユーザ名によって表される），引用（RT という文字列に続いて他人の発言を記述することで表される），ハッシュタグ（シャープ記号の後に発言のトピックを表す単語を書いたもの），URL など，通常の対話システムに用いられることの少ない情報が付与されている事が多い．我々はこれらの情報について正規表現を用いたフィルタリングを行った．

2.3 類似発話対抽出手法

本章では，ユーザ発話と，データベース中の各発話データとの距離を定義する．まずデータベース中にあるデータは 2.2 節で述べたように，Twitter 独自の情報をフィルタリングした後，形態素解析器にかけられる．分析されたデータは品詞タグに基づいてフィルタリングされる．本システムで我々が用いた品詞は名詞，動詞，感動詞である．日本語ツイートには顔文字が多く含まれるため，顔文字に特有な文字を利用したフィルタリングを用いて顔文字の削除も行った．その後，各文書（ツイート）は文書ベクトル $\boldsymbol{x}$ へと変換される．文書 d_i に対して，その文書ベクトルの単語 w_j を表す要素は以下のように表される．

$$x_{i,j} = \frac{tf_{i,j}}{n_j}, \tag{1}$$

ここで，$tf_{i,j}$ は単語 w_j が文書 d_i 中に現れた回数を，n_j は文書 d_i の長さを表す．

文書 $\boldsymbol{x}_a$, $\boldsymbol{x}_b$ 間の類似度は，文書ベクトル同士の内積

$$Similarity(d_a, d_b) = \boldsymbol{x}_a^{\mathrm{T}} \boldsymbol{x}_b. \tag{2}$$

を用いて表される．

2.4 リアルタイムクラウドソーシング

我々はこの対話システムを「リアルタイムクラウドソーシング」と統合する枠組みを提案する．対話システムがユーザ発話に対して適切な応答をデータベース中から見つけられなかったとき，すなわちデータベース中でユーザ発話と最も近い発話の，ユーザ発話に対する類似度がある一定の閾値を超えなかったとき，システムはそのユーザ発話を他のユーザ群（クラウド）に対して行う（クラウドソースする）．もしクラウドの一人がその発言に対して応答をした場合，システムはその応答を用いて元のユーザに対して返答する．一方で，一定時間クラウドから返答がなかった場合，システムはデータベース内でユーザ発話と最も近い発話と判定された発話の応答をシステムの返答とする．この仕組みのひとつの利点は，クラウドソーシングの仕組みの巧妙な隠蔽化である．クラウドであるユーザは自身がクラウドであることを認識せず，単にシステムから語りかけられたと認識する．もちろんはじめにユーザ発話を行ったユーザも，クラウドの存在は意識せず，単にシステムと会話していると感じるのみである．また，本システムは簡潔かつ汎用的な実装という利点も合わせ持つ．システムは，有効な発話がデータベース中に見つからなかった場合，ユーザ発話を利用して新規ユーザ（クラウド）と会話を開始するだけである．クラウドから返答があればそれを，またタイムアウトに達すればデータベース中でもっとも類似度が高かった返答候補を元のユーザに返答する．このクラウドソースの枠組みは，ユーザに対して話しかけることができ，かつ応答生成の際にその応答の適切さをスコアとして算出できるシステム全てに適応することができる．本システムでは式 2 で表される類似度をスコアとして用いた．

さらに，提案システムにおいてデータベース中に適切なデータが蓄積されていない場合に人の助けを借りることができるということは，システムの信頼性を高めるとともに，システムのデータ蓄積効率を高めることができる．

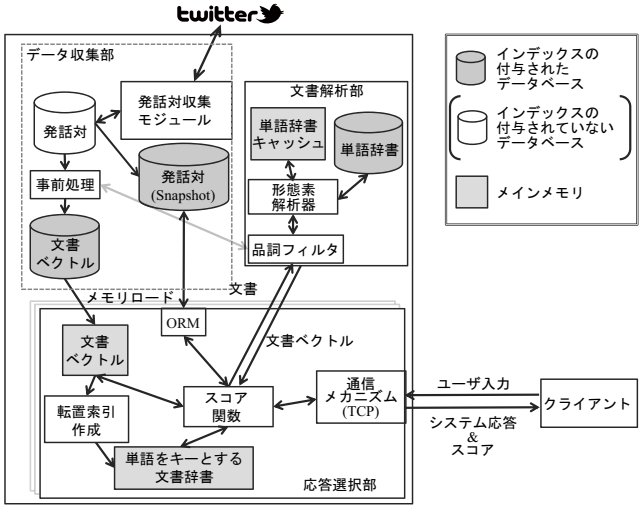


図 3 システムの実装
Fig. 3 System Implementation.

2.5 実装

クラウドソースの枠組み以外のシステム実装を図 **3** に示す．ユーザがクライアントを用いて発話をすると，システムは応答を返す．システムは 3 つの部分からなる．す

[*3] https://dev.twitter.com/docs/streaming-api
[*4] Twitter サービスにおいて日本語を言語として設定しているユーザが書いた投稿はすべて日本語であると仮定した．
[*5] https://dev.twitter.com/docs/api

Vol.2012-NL-206 No.13
Vol.2012-SLP-91 No.13
2012/5/11

表 2　データ収集のための計算・実行速度
Table 2　Calculation speed for Data Collector.

	ツイート/日
ツイッターサンプリング (Streaming API)	7.5×10^5
（内日本語，かつ in-reply-to 情報の含まれたもの）	3.9×10^4
応答元の収集 (REST API)	9.0×10^4
文書ベクトルの計算	8.6×10^6

なわち，データ収集部，文書解析部，回答選択部である．データ収集部においては 2.2 節で述べた方法でシステムは Twitter から対話データの収集を行う．同時に事前処理として，各文書に対して文書ベクトルの計算も行う．文書解析部は形態素解析機を利用して文書をベクトルへ変換する．品詞フィルタリングと，顔文字の除去はこの部分で行う．形態素解析器には MeCab 0.98[*6]を用いた．回答選択部はサーバとして実装される．まず文書ベクトルはメモリにロードされ，転置インデックス（単語がキーとなり，その単語が含まれる文書一覧を返す辞書）が作成される．式 2 より，ユーザ発話と単語に重複がない文書全てについて，ユーザ発話との類似度は 0 となることから，この転置インデックスを利用して，これらの文書の計算を省くことができる．システム全体を通じて，データのメモリへのロードとデータベースへのインデックスの付与を行うことで実時間での処理を実現している．

3. 評価

3.1 コーパスの評価

我々のシステムは一日に 39,000 以上の発話対を収集できる（表 **2**）．並列化により，さらなる高速化が可能である．我々の取得したデータにおける発話長分布を図 **4** に示す．長さの対数と頻度の対数はおおよそ線形の関係にあり，これは英語 Tweet において観測される傾向 [10] と等しい．

3.2 類似発話対抽出手法の評価

実際にシステムに対して入力されたことのあるユーザ発話（図 1 中 U）90 個に対し，Twitter 中のデータから作成したデータベース中の発話対（図 1 中 A, B）を各 20 個ずつ，計 1,800 組用意した．30 人の被験者に対し，各ユーザ発話 (U) と，発話対中の応答 (B) の組を 600 組示し，ユーザ発話に対する応答としての適切さについて評価を行った．つまり，各組に対して 10 名が評価を行うことになる．各被験者は各組に対して応答の自然さ，汎用性の評価を行う．ここで，汎用的な応答とは様々な発言に対する応答として適切となる応答のことを言う．例えば，「どういう意味ですか？」という応答は，ほぼすべての発言に対して適切な応答となり得るため，汎用的な応答となる．[8] はチャットボットが人を騙すためのトリックを多く指摘しており，汎用的な応答もそのひとつと考えられる．例えば，PARRY[3] は無知の導入（「知りません」など）や会話のトピックの変更（「なぜそのようなことを聞くのですか？」など）を用いている．我々のシステムでは大規模なデータベースを用いることで，この汎用的な応答を返すことを避けることを目指す．本研究では，10 名中 8 名以上が自然と判定し，かつ 10 名中 6 名以下が汎用的と判定した応答を正例，それ以外を負例としてシステム評価のための教師データを作成した．

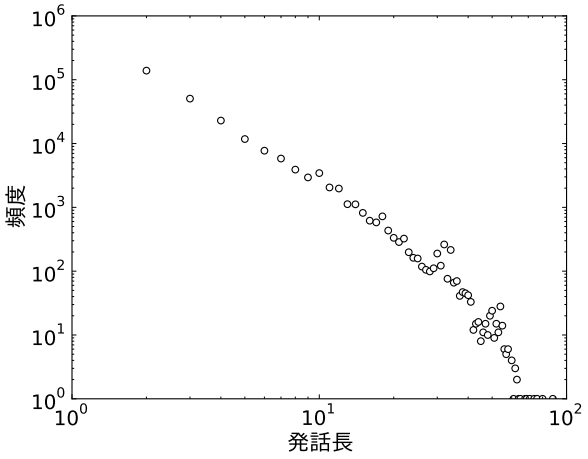


図 4　Twitter の発話長分布
Fig. 4　Twitter utterance length distribution.

ここからは，正例・負例として分けられた教師データを利用してシステムの評価する方法について述べる．システムはデータベース中の応答のユーザ発話に対する適切さをスコアとして計算する．そこで，このスコア関数を識別器とみなすことを考える．つまり，ユーザ発話と発話対を入力したときに，スコア関数が計算する値がある閾値を超えれば発話対の応答は適切（正例）と判定し，そうでなければ不適切（負例）と判定する．

まず ROC カーブについて説明を行う．ROC カーブと，ROC カーブを用いて計算される AUC（後に説明する）を用いることで，確率やスコアの形で出力を行う識別器の評価を行うことができる．ROC 空間は陰性者を間違えて陽性と判定する確率である偽陽性率 (FPR: false positive rate) と陽性者を正しく陽性として判定する確率を表す真陽性率 (TPR: true positive rate) をそれぞれ X 軸，Y 軸にプロットすることによって表される．なお，真陽性率は敏感度 (sensitivity) と呼ばれることが多いが，直感的ではないため，本論文に置いては真陽性率という表現で統一する．

分母も含め，式の形で表現したものを式 3, 4 に示す．

[*6] http://mecab.sourceforge.net/

$$r_{FPR} = \frac{\text{Negatives incorrectly classified}}{\text{Total negatives}}, \quad (3)$$

$$r_{TPR} = \frac{\text{Positives correctly classified}}{\text{Total positives}}. \quad (4)$$

通常の二値識別器，つまりあるサンプルに対して陽性か陰性かを判定する識別器は ROC 空間上の点で表される．例えばすべてのサンプルに対して正しく識別する識別器は偽陽性率が 0，真陽性率が 1 なので，ROC 空間上の点 $(0, 1)$ で表される．一般に，ROC 上で左上に位置する識別器ほど性能が良い識別器と言える．

しかし，入力に対して確率やスコアの形で出力を行う識別器はそのままでは ROC 上にプロットを行うことができない．例えば，ある入力に対して陽性である確率を出力する識別器がこの例として考えられる．そこで，この識別器に対して例えば前述の確率が 0.8 以上である場合は識別器は入力サンプルを陽性と，そうでなければ陰性と判定したとすることにより，この識別器を前述の二値識別器として扱うことができる．つまり，この識別器の性能を ROC 空間上の 1 点で表すことができる．さらに，閾値を連続的に変化させ，各場合について ROC 空間上にプロットし線で繋ぐことで，出力が確率であるこの識別器の性能を閾値の値によらない曲線で表せる．この曲線のことを ROC カーブと呼ぶ．性能が高い分類器ほど，ROC カーブは左上に膨らんだものとなり，全くランダムな識別器においては点 $(0, 0), (0, 1)$ を結ぶ直線となる．この曲線の下の面積を AUC (Area Under the ROC Curve) と呼び，識別器の性能の比較のために用いられる．ランダムな識別器の AUC は 0.5 であり，理想的な識別器の AUC は 1.0 である．この AUC が高いほど一般的にはよい識別器であると言える．

本論文では，文書同士の近さをスコアとして算出する関数を識別器として用いる．スコアの算出は基本的に式 2 に従い，近さを算出する文書の種類や，文書ベクトルの算出方法を変えながら，それぞれの識別器の AUC の値を比較することで各識別器の評価を行う．

比較のために用意した識別器は，それぞれ以下の 6 つの要素について 2 つの選択肢のいずれかを選択することによって区別される．つまり我々は合計で 64 $(=2^6)$ 個の識別器について評価を行った．

- 文書同士の距離比較の際に図 1 中 A のみを考慮するか，もしくは A と B のどちらも考慮するか．
- 文書ベクトル算出の際に単語出現頻度（式 1 中 tf）を考慮するかどうか．
- 逆文書頻度 (idf) を考慮するかどうか．
- Twitter に特徴的な表現を削除するかどうか (2.2 節)．
- 文書ベクトルを文書長によって正規化するかどうか．
- 品詞によるフィルタリングを行うかどうか．

図 5 に結果の一部を載せる．Twitter に特徴的な表現の削除は識別器の性能に大きく寄与しない．むしろ文書正規化や品詞によるフィルタリングが分類器の性能向上に寄与することが分かる．tf，idf の考慮に関して，tf の効果は他の要素によって変わり，識別性能は向上も低下もしうる．idf は性能を低下させる傾向がある（図 5 中 c, d, g, h）．

我々は，AUC の値が最も高かった (0.803) 識別器を，本システムにおける応答選択手法として選択した．詳細を**表 3** に載せる．

表 3　我々が選択したスコア関数
Table 3　Scoring function we choose.

距離計量をする際の文書は？ (A or A+B)	A+B
tf を使うか？	YES
idf を使うか？	NO
Twitter 特有表現を削除するか？	YES
品詞フィルタリングをするか？	YES

表 4　システム計算時間リスト（データサイズ = 1,154,621）
Table 4　System calculation time list. (datasize = 1,154,621)

サーバ起動時間	**433** 秒
文書ベクトルのメモリロード	61 秒
発話対データベースのメモリロード	217 秒
転置索引の計算	86 秒
その他	69 秒
ユーザ入力に対する平均応答時間	**2.57** 秒

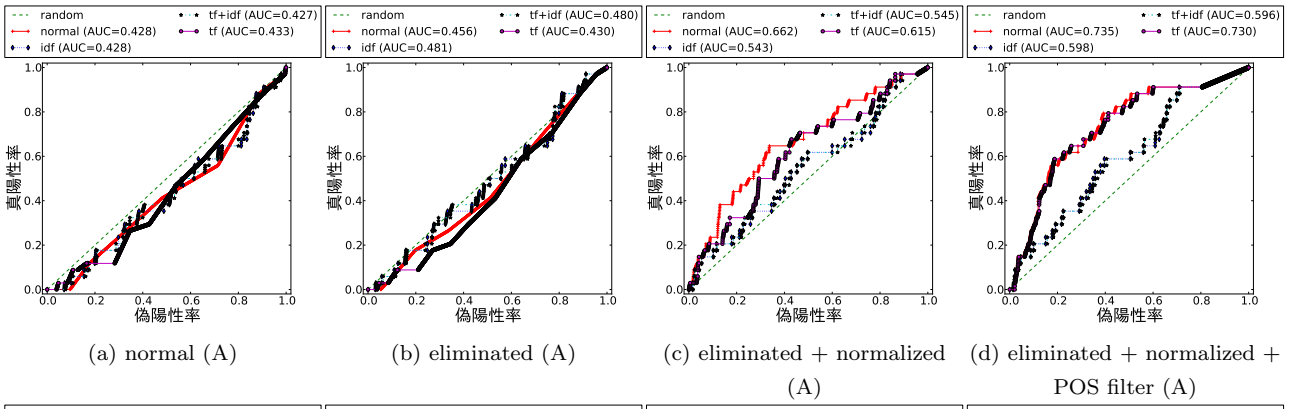
3.3　システムの実装

我々はすべてのシステムを Python で実装し，Intel Xeon Processor X5355 (2.66 GHz, 4 core) と 24GB のメモリを搭載したワークステーションで運用を行った．システムは実行時に 3.2GB のメモリを消費した．なお，CPython の仕様上の制約 (Global Interpreter Lock) からこのサーバが基本的に 1 スレッドでの動作となっていることを付け加えておく．システムの計算時間を**表 4** に示す．**図 6** にシステムの応答時間の分布を示す．システムの応答時間の分布は二峰となっていることが分かる．もしユーザからの発話が一般的な（データベース中に広く含まれる）単語を含む場合は，システムはデータベース中のほとんどの発話対に対してユーザ発話との距離を測る必要性が生じる．これが応答時間分布の右側のピーク (5-9 秒あたり）の場合を表す．しかし，図よりほとんどの場合においてシステムは 2 秒以下での応答をしており，これはユーザの想定する応答時間の制限に適する（表 1)．また，ユーザ入力とデータベース中の発話対との距離の計算は並列に行うことが可能であり，分散処理を行うことによってより大きなデータサイズにも対応可能であると考えられる．

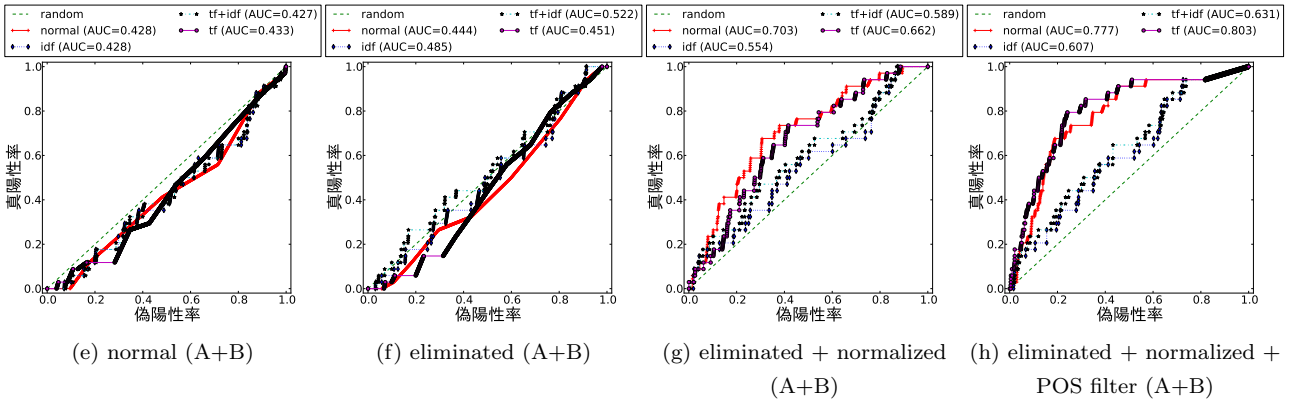
4.　実験

4.1　データサイズ

システムのデータサイズの違いが挙動の違いに与える影



(a) normal (A) (b) eliminated (A) (c) eliminated + normalized (A) (d) eliminated + normalized + POS filter (A)



(e) normal (A+B) (f) eliminated (A+B) (g) eliminated + normalized (A+B) (h) eliminated + normalized + POS filter (A+B)

図 5 それぞれの分類器における ROC カーブ．各図には tf，idf それぞれの適用有無によって 4 本の線が書かれている．上段には図 1 中の A のみが考慮された場合が (a, b, c, d)，下段には A と B の両方が考慮された場合が (e, f, g, h) 示されている．Normal, eliminated, normalized, POS filter はそれぞれ，何もしないこと，Twitter 特有表現の削減，文書長での正規化，品詞フィルタリングを表す．

Fig. 5 ROC curve for each scoring function. In each graph there are 4 lines, and each line represents whether tf and idf are used to calculate the document vector. Only A in Figure 1 is treated in the first line (a, b, c, d), whereas A and B is considered in the bottom (e, f, g, h). Normal, eliminated, normalized, POS filter mean doing nothing, twitter-specific description is eliminated, normalized by character count, considering only specified POS, respectively.

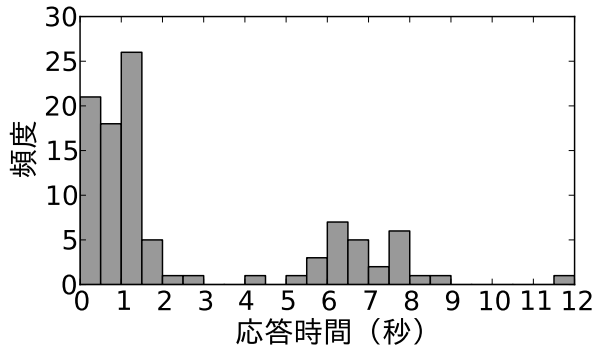


図 6 システム応答時間分布

Fig. 6 System response time distribution. (datasize = 1,154,621)

響を調べるために，システムの持つデータサイズを変えながら，コンソール上で被験者に 3 分間の会話をシステムと行ってもらう被験者実験を行った．被験者はシステムと会話した後，応答の自然さと会話の面白さについてアンケートで評価を行う．データサイズは 100, 1k, 10k, 100k, 500k の 5 種類を用意した．結果は図 **7**, 8 に示す．

データサイズが 100 から 100k に上がる過程においてはシステムに対する被験者の評価は上昇する傾向にあるが，データサイズが 100k から 500k へと増えた場合，システムに対する評価は有意に下がっている．我々は，この傾向が 3.2 節で述べた選択手法の違いによって現れて可能性について探った．具体的には表 3 の最上段に示した距離計量をする際の文書を A のみにしたものを用い，同じ条件の元実験を行った．ただし，データサイズについては 100 万に増やしたものも用意した．図 **9**, 10 に示された結果は，本傾向が異なる条件に置いても起こることを示している．さらに，データ数が 100 万に上がると，500k の時に比べて性能の向上が起こることも示している．

Vol.2012-NL-206 No.13
Vol.2012-SLP-91 No.13
2012/5/11

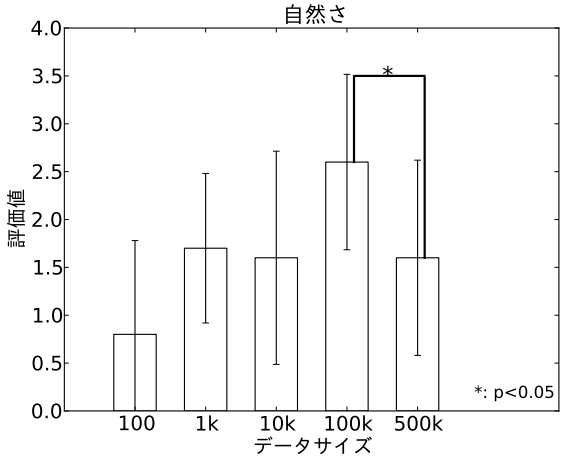


図 7　データサイズの違いによるシステム応答の自然さ評価
Fig. 7　Evaluation score versus data size (naturalness).

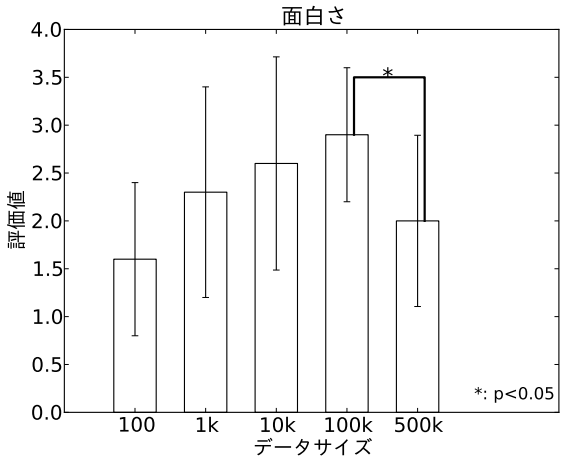


図 8　データサイズの違いによるシステムの面白さ評価
Fig. 8　Evaluation score versus data size (interesting).

4.2　リアルタイムクラウドソーシング

最後に我々は提案するクラウドソーシングの枠組みを加えて実験を行った．我々は Twitter クローンである StatusNet[*7]をインターフェースとして用いた（図 **11**）．最初我々は Twitter をインターフェースとして用いることを試みたが，数分で一日の投稿制限を超えてしまったため断念した．比較実験とするため，クラウドソーシングを除いた枠組みでも実験を行った．被験者数は 10 に，クラウドサイズは 3 に設定した．つまりユーザからの入力に対してシステムが有効な応答をデータベース中から発見できなかった場合にはシステムが他の 3 人のユーザに対してその応答を投げかけるということである．実験の後，各被験者はシステム応答の自然さと，システム対話で感じた面白さについてアンケートに回答した．結果を図 **12** に示す．驚いたことに自然さについては被験者はクラウドソーシングの枠組みがないシステムに対して若干高い評価をつける結果となった．一方で，クラウドソーシングの枠組みの追加は被験者に「面白さ」を与えるという結果になった．この実験において被験者はテニスの話題を話すことがあった．なぜなら，当時日本人テニスプレーヤが世界の大きな大会において快進撃を続けていたからである．そして，データベース中にはテニスの話題が少なかったことからこの話題は他のユーザにクラウドソースされ，クラウドソースされたユーザは突然システムからホットなテニスの話題をふられることとなった．アンケートの感想欄からもこのような例がシステムがユーザに面白さを感じさせるのに寄与したことが確認された．

[*7] http://status.net/

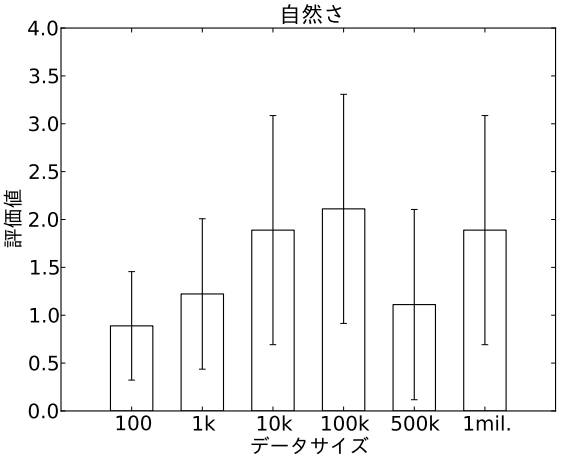


図 9　データサイズの違いによるシステム応答の自然さ評価．ただし距離計量の際システムは図 1 中 A のみしか考慮していない．
Fig. 9　Evaluation score versus data size (naturalness). System only considers similarity between user utterance and A in Figure 1.

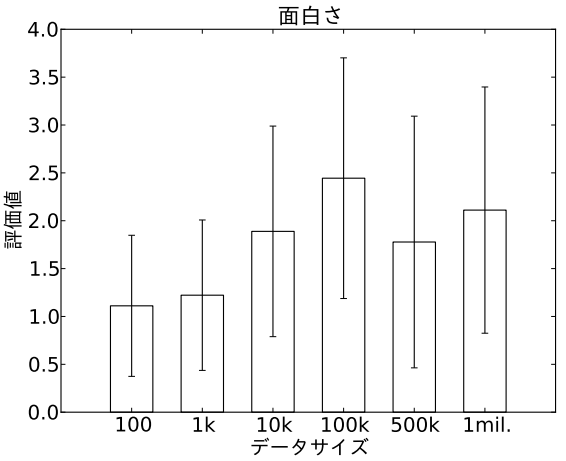


図 10　データサイズの違いによるシステムの面白さ評価．ただし距離計量の際システムは図 1 中 A のみしか考慮していない．
Fig. 10　Evaluation score versus data size (interesting). System only considers similarity between user utterance and A in Figure 1.

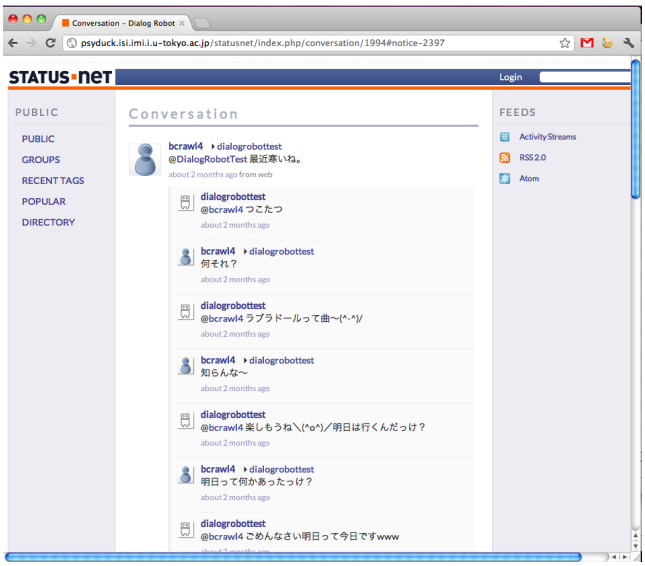


図 11　StatusNet での実装
Fig. 11　System implementation on StatusNet.

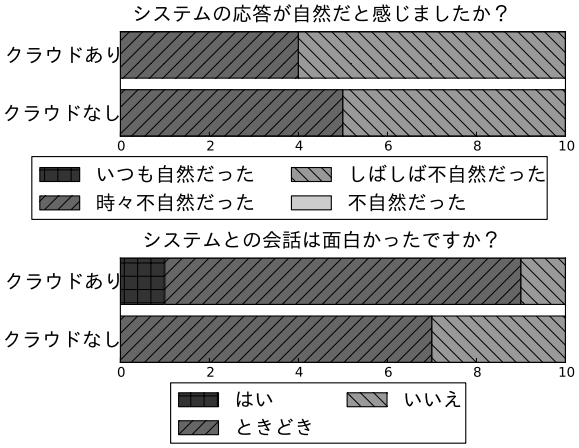


図 12　クラウドソーシングを統合したシステムにおけるアンケート結果
Fig. 12　Survey result of the whole system.

5. 結論と今後の展望

本論文で我々は大規模対話データベースとリアルタイムクラウドソーシングに基づいた新規な対話システムを提案し，実装，実験を行った．我々のシステムはデータベースインデクシングとメモリへのプリロードによって 100 万を超える対話データに対して平均 2.54 秒の応答時間を達成した．実験に置いては多くの場合でデータサイズの増加によって，ユーザがシステムに対して面白さを，またシステム応答に対して自然さを感じる割合が増えることを示した．ただしそうでない場合も観測された．また，クラウドソーシングの枠組みの統合がシステム応答の自然さというよりはシステム全体の面白さの向上に寄与するということがわかった．

今後我々は発話対コーパスの規模を拡大していく予定である．その上で今回と同じ枠組みで同じような実験を行い，データサイズがさらに大きい場合の可能性について探りたい．また，今回は 2 つの発話からなる発話対に注目したが，これを 3 つ以上へと拡張することによって文脈を考慮した対話システムを作ることが可能であると考えられる．我々の解析では，対話形式な日本語のツイートの内 41%以上は発話が 3 つ以上続く発言である．例えばシステムの「私は犬が好きです．」という発言に対してユーザが「なんで？」という応答をしたとする．このとき現在の我々のシステムでは単に「なんで？」に近い応答をデータベースから探そうとするため，「犬」という情報は抜け落ちてしまう．しかし 3 つに拡張した場合はそうではない．

参考文献

[1] E. Bakshy and J. M. Hofman. 2011. *Everyone's an Influencer: Quantifying Influence on Twitter*. In Proc. of International Conference on Web Search and Data Mining.
[2] J. Bollen, H. Mao, and X. Zeng. 2011. *Twitter Mood Predicts the Stock Market*. Journal of Computational Science.
[3] K. Colby. 1975. *Artificial Paranoia : A Computer Simulation of Paranoid Processes*. Pergamon Press, New York.
[4] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith 2010. *Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments*. The Defense Technical Information Center Document.
[5] R. Higashinaka, N. Kawamae, K. Sadamitsu, and Y. Minami. 2011. *Building a Conversational Model from Two-Tweets*. In Proc. of IEEE Workshop on Automatic Speech Recognition and Understanding.
[6] L. Jiang, M. Yu, M. Zhou, X. Liu, and T. Zhao. 2011. *Target-dependent Twitter Sentiment Classification*. In Proc. of Annual Meeting of the Association for Computational Linguistics: Human Language Technologies.
[7] R. Kelly. 2009. *Pear Analytics Twitter Study (Whitepaper)*.
[8] M. L. Mauldin. 1994. *Chatterbots, TinyMuds, and the Turing Test Entering the Loebner Prize Competition*. In Proc. of the National Conference on Artificial Intelligence.
[9] D. Ramage, S. Dumais, and D. Liebling. 2010. *Characterizing Microblogs with Topic Models*. In Proc. of the International AAAI Conference on Weblogs and Social Media.
[10] A. Ritter, C. Cherry, and B. Dolan. 2010. *Unsupervised Modeling of Twitter Conversations*. In Proc. of North American Chapter of the Association for Computational Linguistics - Human Language Technologies.
[11] Richard S. Wallace. 2009. *The Anatomy of A.L.I.C.E.*. Parsing the Turing Test.