PodCastle：ポッドキャストをテキストで検索，閲覧，編集できるソーシャルアノテーションシステム

PodCastle: A Social Annotation System Where Podcasts Can Be Searched, Read, and Annotated by Text.

緒方 淳　　後藤 真孝　　江渡 浩一郎*

概要.　本稿では，音声認識を用いて Web 上のポッドキャストの全文テキスト検索を可能にするソーシャルアノテーションシステム「PodCastle」を提案する．PodCastle のユーザは，検索語を含むポッドキャストを見つけて聞けるだけでなく，検索したポッドキャストの自動書き起こしテキストを読んで内容を確認することができる．さらに，書き起こしテキスト中に音声認識誤りを見つけたら，自動表示される他候補を選択するだけで容易に訂正 (アノテーション) することもできる．PodCastle は，こうして多数のユーザが訂正を繰り返した正解情報を自動学習することで，音声認識器の性能を徐々に向上させられる特長を持つ．あらゆるポッドキャストを正しく書き起こせるような既存の音声認識器は存在しないが，この性能向上の仕組みによって音声認識器が徐々に強力になって認識誤りが減ることで，PodCastle の有用性も高まっていくことが期待できる．

1　はじめに

音声の検索は難しい．なぜなら，検索に必要な索引情報 (文やキーワード等) を，音声から抽出することが困難だからである．一方，テキストの検索は既に広く使われており，Google 等の優れた検索エンジンにより，Web 上のテキストを含む各種ファイルに対する全文検索が可能となっている．もし Web 上の音声ファイルからその発話内容のテキストを抽出できれば，同様に全文検索が可能になるが，一般に様々な内容に対してそうした音声認識を実現しようとすると，認識率が低くなる．そのため，Web 上に音声ファイルが多数公開されていたとしても，特定の検索語を含む発話へピンポイントにアクセスするような全文検索は難しかった．

しかし近年，音声版のブログ (Weblog) ともいえる「ポッドキャスト」が普及し，Web 上の音声ファイルとして多数公開されるようになったため，そうした音声に対する全文検索の重要性がより一層増してきた．そこで，英語のポッドキャストに対して音声認識を利用して全文検索を可能にするシステム「Podscope」[1],「PodZinger」[2] が 2005 年から公開され始めた．いずれも，音声認識によりテキスト化した索引情報を内部に持ち，ユーザが Web ブラウザ上で入力した検索語を含むポッドキャストの一覧を提示される．Podscope では，ポッドキャストのタイトルだけが列挙され，検索語が出現する直前から再生できるが，音声認識されたテキストは一切表示されない．一方，PodZinger では，これに加え，検索語が出現した周辺のテキスト (音声認識結果) も表示され，ユーザがより効率的に部分的な内容を把握できるようになっている．しかし，せっかく音声認識をしていても，表示されるテキストは一部に限定されており，音声を聞かずにポッドキャストの詳細な内容を視覚的に把握することはできなかった．また，音声認識では認識誤りを避けることはできないため，ポッドキャストに対して誤った索引付けがなされて検索に悪影響を与えるが，それをユーザが把握したり改善したりすることは不可能だった．

そこで本研究では，Web 上の日本語のポッドキャストを音声認識によって自動的にテキスト化することで，それらをユーザが全文検索できるだけではなく，詳細な閲覧，編集も可能なシステム「PodCastle」を提案する．PodCastle では，検索したポッドキャストの全文をテキスト表示することで，音声再生環境がなければ内容を把握できないポッドキャストを「読む」ことも可能にする．従来こうしたシステムが実現困難だったのは，ポッドキャストの多様な音声に対して，高い音声認識率を達成することが難しかったからである．本研究では，これを解決するために，システムが持つすべての情報を積極的にユーザに開示し，多数のユーザに認識誤りを訂正 (認識結果を編集) する協力をしてもらうことで，音声認識率をシステムの運用中に向上させる枠組みを提案する．これにより一般ユーザだけでなく，作成者 (ポッドキャスタ) が自身のポッドキャストの認識誤りを把握し，訂正して適切に検索されるように改善できる．

そうしたユーザの自発的な協力を得るために，PodCastle では音声認識結果のテキスト全文を表示するだけでなく，認識誤りを容易に訂正できる編集 (アノテーション) 機能を提供する．閲覧中に認識誤りを

* Jun Ogata, Masataka Goto, and Kouichirou Eto 産業技術総合研究所

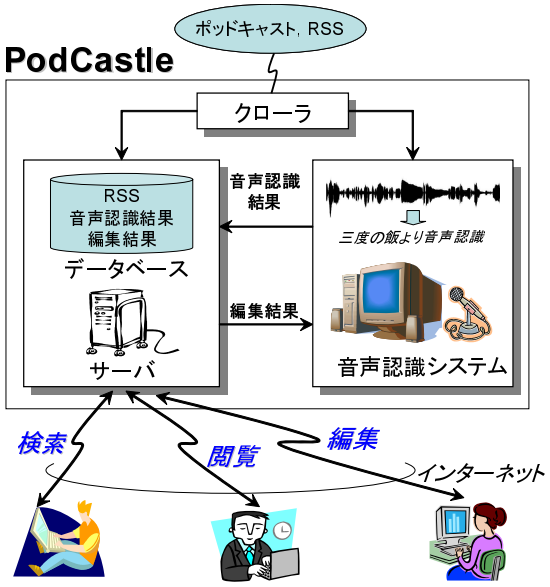

図 1. PodCastle システム

ユーザが発見してマウスやキーボードで指摘すると，その競合候補 (音声認識中に，最終的な認識結果以外に可能性の高かった単語候補) が自動的に表示される．ユーザはその周辺の音声を聞きながら，正しい候補を選択するか，正しいテキストをタイプすることで訂正する．これにより単にテキストエディタで訂正するのに比べ，効率が良く，気軽に作業ができる．以上のシステムを Web 上で公開して，多数のユーザが検索，閲覧，編集を繰り返すと，誤認識箇所に関する正解情報が日々集積されるので，それを自動学習することで音声認識性能も向上できる．つまり，個々の訂正の影響はそのポッドキャストにとどまらず，未訂正のポッドキャストに対する性能向上も可能となる．こうした多数のユーザによる「ソーシャルアノテーション」を利用したアプローチは，従来の音声認識研究では実現されていなかった新たな性能向上の枠組みと言える．

以下，2 章において PodCastle が実現する機能の特長を議論し，3 章で各機能の具体的な操作を説明する．4 章では実現方法とその具体的な実装について述べ，5 章で PodCastle の小規模な運用結果について述べ，最後に 6 章にてまとめを述べる．

2 PodCastle の概要と特長

本研究で提案する「PodCastle」は，ポッドキャストを対象としたソーシャルアノテーションシステムである．近年，Web 上の音声データをダウンロードし，iPod などのデジタルオーディオプレーヤーで効率良く聞く仕組み「ポッドキャスティング」が普及した．ポッドキャストとは，そこで配信される音声データ (MP3 ファイル) とそのメタデータのことで，その流通を促すために，ブログなどで更新情報を通知するために用いられているメタデータ RSS (Really Simple Syndication) 2.0 が必ず付与されている点が，単なる音声データと違う点である．この仕組みにより，ポッドキャストは音声版ブログともいわれ，個人による音声データの発信，流通，入手が容易にできるため，Web 上では今後，音声データが増加の一途をたどると予測される．したがって，Web 上のテキストデータの場合と同様に，PodCastle によってポッドキャストに対しても全文検索や詳細な閲覧が可能になれば，ポッドキャスト同士あるいはテキストデータとの間で構造化が進むことが期待され，その意義は大きい．

図 1 に PodCastle のシステム構成を示す．まず，クローラ (アグリゲータ) により，Web 上のポッドキャスト (音声データと RSS) が収集される．収集された音声データに対して，音声認識器が自動認識処理をし，その認識結果は PodCastle のデータベースに保存される．この認識結果は，通常の音声認識結果 (1 つの単語列) だけでなく，各単語の開始時間と終了時間やその区間の複数の競合候補，信頼度等，再生や訂正に必要な豊かな情報も含んでいる．PodCastle のサーバは，データベース中のこれらの認識結果と収集された RSS を用いることで,「検索」,「閲覧」,「編集」の 3 つの機能をインターネットを通じてユーザに提供する．以下，これら 3 つの機能の特長を述べる．

2.1 「検索」機能

PodCastle では，初めて日本語のポッドキャストに対する全文検索を実現する．しかも，音声認識結果 (およびユーザがそれを編集した結果) の全文テキストを，ポッドキャストに対する単なる索引情報として内部利用するのではなく，1 つの独立したアノテーションとして外部公開する．そのため，PodCastle 上の独自の検索エンジンからだけではなく，外部の一般的なテキスト検索エンジンからの検索も可能になる．

これは意図的に設計した重要なアプローチである．こうすることで，ユーザがテキスト検索エンジン上で全文検索をする際に，通常の Web ページと同時に，その検索語を含むポッドキャストも発見される．その結果，ポッドキャストがより多くのユーザに広まって利便性や価値が高まり，ポッドキャストによる情報発信をさらに促すことができる．これは，作成者（ポッドキャスタ）にとってもメリットがあるので，後述の「編集」機能で訂正する動機付けの一つとなる．

なお，前述した従来システム Podscope，PodZinger では，英語のポッドキャストに対する全文検索は実現されていたが，音声認識により索引付けされた結果は内部に隠蔽され，各システム上でしか検索ができなかった．仮にそれらのシステムが日本

語に対応したとしても，本研究との違いは大きい．音声認識結果の開示による外部の検索エンジンからの利用の重要性を指摘し，それを可能にしたのは，本研究が初めてである．

2.2 「閲覧」機能

ユーザが，検索したポッドキャストを「聞く」だけでなく,「読む」ことも可能にする．これは，音声再生環境がなくても内容を把握したいときに有効であるが，普通にポッドキャストを再生しようとしている場合でも，それを聞くべきかどうか事前に吟味することができて便利である．

ポッドキャストの音声再生は魅力的である一方，音声であるために，その内容に関心があるかどうかを聞く前に把握できなかった．再生スピードを上げることで聞く時間を短縮するにも，限界がある．「閲覧」機能により，聞く前にざっと全文テキストを眺められることで，その内容に関心があるかどうかをより短時間で把握でき，ポッドキャストの取捨選択が効率良くできる．また，収録時間の長いポッドキャストのどの辺に関心のある部分があるのかもわかる．仮に音声認識誤りが含まれていても，こうした関心の有無は充分判断でき，本機能の有効性は高い．

2.3 「編集」機能

PodCastle において最も特徴的なのは，この「編集」機能により，サーバにアクセスしているユーザが誰でも，ポッドキャストに対して「アノテーション」を行える点である．ここでのアノテーションは，ポッドキャストに対して正確な書き起こしテキストを作成することを意味し，音声認識結果中の認識誤りを訂正する形で行われる．ユーザが訂正した結果 (編集結果) は，PodCastle のデータベースに蓄積され，その後の検索機能や閲覧機能に反映されるとともに，音声認識器に送られる．音声認識器側では，その編集結果に基づいて，性能向上のための再学習が行われる．

このように，PodCastle は，システム内部のデータを最大限に公開しつつ，不特定多数のユーザに参加してもらいながら意識的なフィードバックを得て，それらを音声認識の性能向上に利用する．最終的に検索や閲覧などの提供サービスの向上に反映されて，ユーザの積極的な参加を促すことから，Web 2.0 的な「ソーシャルアノテーションシステム」と捉えられる．

3 PodCastle の機能とインタフェース

PodCastle の 3 つの機能を提供するためのインタフェースについて，それぞれの操作を効率的に行うための工夫も交えて述べる．なお，以下ではマウス操作の場合を説明するが，キーボードでも同等の操作が可能である．

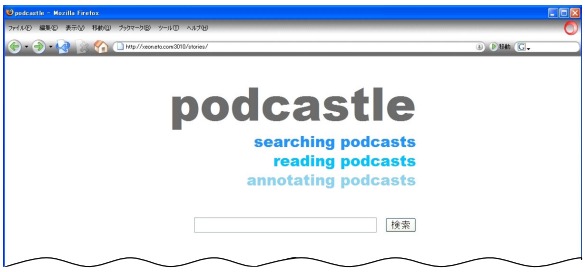

図 2. PodCastle のトップページ

図 3. 検索機能による検索結果画面の例

3.1 検索機能のインタフェース

PodCastle の Web サイトのトップページ (図 2) もしくは検索結果画面 (図 3) の上部から，ユーザがテキストで検索語を入力して検索ボタンをクリックすると，ポッドキャストの認識結果 (アノテーション結果) が全文検索される．そして，図 3 の検索結果画面に切り替わって，その検索語を含む複数のポッドキャストが列挙される．それらは，より適切な結果が上位に出るよう，音声認識の際に求めた単語 (ここでは検索語と同一の単語) の信頼度の高い順に並べられる．

図 3 のように，各ポッドキャストについて，そのタイトルと元の音声ファイルへのリンクだけでなく，検索語が出現した付近のテキストも表示される．また，左側に試聴のための「再生」「停止」ボタンも提供され,「再生」ボタンをクリックすると検索語の周辺の音声を聞くことができる．音声の再生に同期して，カーソル (ハイライト) がテキスト上をリアルタイムに移動し，検索語の出現箇所 (あるいは誤認識された箇所) を的確に把握できる．

そして，あるポッドキャストを閲覧あるいは編集したくなったら，テキスト中の検索語 (リンクが設定されている) か,「全文表示」リンクをクリックすると，次節の閲覧機能に移行する．

図 4. ポッドキャストの全文テキスト閲覧機能，兼，編集機能の全文モードの画面表示例

図 5. 編集機能の詳細モードの画面表示例

3.2 閲覧機能のインタフェース

図 4 のような画面で，ポッドキャストの全文テキストを閲覧し，内容を詳細に把握できる．音声の再生も可能であり，検索結果の画面と同様に，再生と同期してカーソルが移動するため，長いポッドキャストでも再生位置を見失うことがない．画面上方の「再生」ボタンを押すと現在のカーソル位置から再生されるが，全文テキストの各単語も再生ボタンとなっており，ユーザは聞きたい箇所の単語をクリックするだけで，そこから再生できる (ただし，「自動再生」ボタンが押されて有効なときのみ)．単にポッドキャストを聞くだけよりも，全文テキストとその上のカーソルの動きを見ながら再生する方が，内容の理解を促せる．

3.3 編集機能のインタフェース

全文テキストを編集して音声認識誤りを訂正するために，「全文モード」と「詳細モード」の二種類の画面・操作方法を用意した．

全文モードでは，閲覧機能と同じ画面 (図 4) 上で，訂正作業をする．閲覧機能と編集機能はシームレスに使用可能で，音声を聞きながらポッドキャストの閲覧をしていて認識誤りを発見したら，即座に訂正できる．編集機能はその誤り箇所 (単語) をクリックすると始まり，クリックした単語の下に競合候補 (その区間の認識結果として可能性の高かった候補) のリストが表示されるので，ユーザはその中から正しい単語をクリックして選択すればよい．候補中に正しい単語がない場合は，キーボードを用いて自由にタイプして訂正できる．なお，各単語は，音声認識の信頼度に基づいて赤色で着色されており，赤くなっている場所は誤っている可能性が高い．

詳細モードでは，図 5 に示す画面に切り替わり，横一列に並んだ認識結果の各単語区間の下に，それぞれの競合候補のリストが表示される (これは文献 [3] の「音声訂正」と等価な機能である)．このように競合候補が常に表示されているため，誤り箇所をクリックして候補を確認する手間が省け，正しい単語を次々と選択するだけで訂正できる．この表示で，競合候補の個数が多い箇所は認識時の曖昧性が高かった (音声認識器にとって自信がなかった) ことを表しており，候補の個数に注意しながら作業することで，誤り箇所を見逃しにくい．各区間の競合候補は信頼度の高い順に並んでおり，通常は上から下へ候補を見ていくと，早く正解にたどり着けることが多い．また，競合候補には必ず空白の候補が含まれる．これは「スキップ候補」と呼ばれ，その区間の認識結果をないものとする役割を持つ．つまりこれをクリックするだけで，余分な単語が挿入されている箇所を容易に削除できる．

二種類のモードは，図 4，図 5 の画面左上にあるタブをクリックすることで，訂正中のカーソル位置を保存したまま自由に切り替えられる．全文モードは，テキストの閲覧が主目的なユーザにとって有用であり，普段は閲覧の邪魔にならないように競合候補は見えないが，ユーザが認識誤りに気付いたときに，そこだけ気軽に訂正できる利点がある．一方，編集モードは，認識誤りの訂正が主目的なユーザにとって有用であり，前後の競合候補やそれらの個数も見ながら，見通し良く効率的な訂正ができる利点がある．このように，そのときどきの気分で切り替えて，気軽に部分的な訂正が可能である．

4 PodCastle の実現方法

PodCastle を実現するためには，高性能な音声認識器と，編集 (アノテーション) のための効率的な訂正インタフェース，Web サーバとクライアントの一連の機能を実装する必要がある．

4.1 音声認識器の実装

信頼度付き競合候補 (後述する confusion network) を生成できる機能を持つ，独自に開発した大語彙連

続音声認識器を用いた．このデコーダは，back-off 制約 N-best 探索アルゴリズム [4] に基づいている．多様なポッドキャストに対応するために，音響モデルと言語モデルは，以下のような工夫をして高精度化した．

4.1.1 音響モデルの高精度化

音響モデルには，日本語話し言葉コーパス (CSJ) から学習した triphone モデルを用いた．しかしポッドキャストの場合，音声が収録されているだけでなく，背景に音楽や雑音を含む場合がある．そうした音声認識が困難な状況に対処するために，雑音抑圧手法 ETSI Advanced Front-End[5] を用いて，学習と認識の前処理の音響分析を行い，性能を改善した．

4.1.2 言語モデルの高精度化

言語モデルには，CSRC ソフトウェア 2003 年度版 [6] の中から，1991 年から 2002 年までの新聞記事テキストより学習された 60000 語の bigram を用いた．しかしポッドキャストの場合，最近の話題や語彙を含むものが多く，学習データとの違いからそうした音声を認識することが難しい．そこで，日々更新されている Web 上のニュースサイトのテキストを，言語モデルの学習に利用して，性能を改善した．

具体的には，総合的な日本語ニュースサイトである Google ニュースと Yahoo!ニュースに掲載された記事のテキストを毎日収集し，学習に用いた．現在の実装 (および 5 章の実験) では，まず，2006 年 8 月 1 日から 27 日までの間に収集した全テキストだけから，新たな言語モデルを作成した．そして，そのモデルと上述の CSRC ソフトウェアの言語モデルとの間でモデルの融合 [7] を行うことで，最近の話題や語彙に対する適応化を行った．

4.2 編集機能の訂正インタフェースの実装

編集機能の訂正インタフェースは，我々が以前提案した「音声訂正」[3] に基づいている．一般に，音声認識の内部状態には中間候補が膨大にあり，適切な個数に競合候補をまとめ上げるのは難しいが，音声訂正では，confusion network と呼ばれる音声認識の内部状態を簡潔にまとめた中間表現 (信頼度付き競合候補) を，誤り訂正インタフェースへと初めて応用することにより，これを実現した．本研究でも confusion network を用いることで，ポッドキャスト用の大語彙音声認識においても，効果的な競合候補の提示，訂正を可能にした．

4.3 訂正結果を利用した音声認識の高精度化

ユーザが編集機能で訂正した結果は，音声認識性能を向上させるために様々な方法での利用が考えられる．例えば，音声データ全体に対する正しいテキスト (書き起こし) が得られるので，音声認識の一般的な方法で音響モデルや言語モデルを再学習すれば，性能向上が期待できる．そこで本研究では，これと併用できる，より研究上の意義の大きい試みとして，個々の訂正箇所に着目した手法を提案する．

本手法では，音声認識器が誤りを起こした発声区間が，どのような正解単語へ訂正されたのかがわかるので，その区間の実際の発声 (発音系列) が推定できれば，正解単語との対応が得られる．一般に音声認識では，事前に登録した各単語の発音系列の辞書を用いて認識するが，実環境での音声は予測困難な発音変形を含むことがあり，辞書の発音系列と一致せずに誤認識を引き起こす原因となっていた．そこで，誤りを起こした発声区間の発音系列 (音素列) を音素タイプライタ (音素を認識単位とした特殊な音声認識器) により自動推定し，その実際の発音系列と正解単語の対応を辞書に追加登録する [8]．こうすることで，同じように変形した発声 (発音系列) に対して辞書が適切に参照でき，同じ誤認識を再び起こさないことが期待できる．また，ユーザがタイプして訂正した，事前に辞書に登録されていなかった単語 (未知語) も認識できるようになる．

4.4 Web サーバとクライアント (インタフェース) の実装

一連の機能のサーバ側動作は，Web アプリケーションフレームワーク Ruby on Rails 1.1.4 で実装されている．プログラミング言語 Ruby 1.8.4，Web サーバ WEBrick 1.3.1，データベース MySQL 4.1.21 を用い，ポッドキャストのメタデータと音声認識結果を登録して運用した．ユーザがブラウザ (クライアント) 上で検索語を入力すると，それがサーバ側で 茶筌 2.3.3 によって形態素解析され，MySQL の全文検索機能 (Full Text) によって検索される．

クライアント側のインタフェース (画面描画，カーソル移動，候補表示等) の機能は，JavaScript 1.5 及び JavsScript ライブラリ MochiKit 1.3.1 を用いて実装した．ただし，音声ファイル中の任意の位置からの再生は，JavaScript だけでは実現できないため，FlashProxy 0.3 を用い，JavaScript と Flash プラグインと連携させることによって実現した．再生用 Flash コンポーネントは，ActionScript 2.0 で記述し，MTASC (Motion-Twin ActionScript 2 Compiler) 1.12 でコンパイルして生成した．

5 実験

音声認識の高精度化の評価実験結果と，実装したシステムの運用結果について述べる．

5.1 音声認識の評価

4.1.2 節，4.3 節で提案した高精度化手法の潜在能力を評価するため，小規模ではあるが，実際のポッドキャストを用いた実験を行った．評価データとし

ては,「日刊工業新聞 Podcast ビジネスライン [9]」のポッドキャストのうち，2006 年 8 月 23 日，24 日に公開された 2 つを用いた．

まず，4.1.2 節の言語モデルの適応化の実験結果を表 1 に示す．結果より，Web テキストを用いた適応化を行うことで，未知語数が削減され，認識率も改善されていることがわかる．今回利用したような総合的なニュースサイトでは記事が日々更新され，しかも話題も広範囲なため，ポッドキャストのように話題を特定できない対象に対して，特に有効であるといえる．

次に，4.3 節の訂正箇所を利用した高精度化の実験結果について述べる．ここでは，2006 年 8 月 23 日のデータに対する編集結果を音声認識器に反映し，2006 年 8 月 24 日のデータを用いて認識実験を行った．結果としては，今回のデータに関しては，共通する話題，特定の語彙が存在しなかったため，全体としての認識性能はほとんど変わらなかった．しかし，両データに共通して出現する特定のフレーズ箇所 (「続いてのニュースです」,「担当は〜がお伝えしました」など) については改善が見られた．本手法は，特定の話題が長く続くようなデータや，ある単語が未知語で，その単語が頻繁に出現するようなデータの場合に，特に大きな効果が得られると考えられる．そのようなデータにおける評価は今後の課題である．

表 1. 4.1.2 節の言語モデルの適応化による性能改善

	未知語数	認識率
ベースライン	23 個	78.1 %
適応化 (提案手法)	7 個	85.3 %

5.2 運用結果

小規模ではあるが，ニュース関連のポッドキャストを用いて PodCastle を運用し，4 人のユーザが本システムの 3 つの機能を体験した．その結果，提案した機能が適切に動作することが確認できた．今後は，一般公開を通じてユーザスタディをしていく予定である．

6 まとめ

本稿では，音声認識を活用し，ポッドキャストの自動書き起こしテキストの全文検索，全文テキスト閲覧，テキスト訂正編集を可能にする「PodCastle」という新たなソーシャルアノテーションシステムを提案した．PodCastle を，敢えて単なる検索エンジンと呼ばなかったのは，ユーザが共同で訂正作業をしながら，ポッドキャストの正しい全文テキスト (アノテーション) を作り上げていける点を重視しているからである．実際に，訂正されるごとに音声認識性能も向上していく枠組みなため，ユーザがあたかも「システムを育てている」かのような感覚が得られることも狙っている．これは，システム提供者側が事前に大量のデータベースから音声認識器を学習しておき，それを用いて外部非公開の索引情報を静的に検索する従来の枠組みとは大きく異なり，新規性と透明性が高く発展的なアプローチと言える．

このアプローチの場合，全テキストを外部公開することで，音声認識器の性能が誰の目にも明らかになり，誤認識が多いポッドキャスト等があると我々が批判を受ける可能性はあるが，そうした現状をユーザと共有してはじめて，音声認識技術の真の普及と発展があると我々は信じる．今後 PodCastle が普及すれば，場合によっては，ポッドキャストを収録する時点で，意識的に誤認識されにくいきれいな発声をしようとする試み (一般に音声認識では，人間の聞き取りやすいアナウンサーの発声が一番認識率が高い) や，作成者がポッドキャストの収録原稿を正解テキストとして PodCastle へ直接入力する試みも生まれるかも知れない．

今後は，音声認識性能のさらなる向上と共に，ビデオキャストへの対応や，コミュニティー形成のための各種機能拡張 (ユーザ管理，ランキング等) に取り組んでいきたい．

謝辞

PodCastle の Web サーバとクライアントの実装を担当して頂いた有限会社ブラジル（代表取締役 上津 竜太郎 氏）に感謝する．

参考文献

[1] http://www.podscope.com/

[2] http://www.podzinger.com/

[3] 緒方，後藤: “音声訂正: 認識誤りを選択操作だけで訂正ができる新たな音声入力インタフェース” WISS2004, 論文集, pp.47-52, 2004.

[4] 緒方, 有木: “大語彙連続音声認識における最ゆう単語 back-off 接続を用いた効率的な N-best 探索法”, 信学論, Vol.84-D-II, No.12, pp.2489-2500, 2001.

[5] ETSI ES 202 050 v1.1.1 STQ; distributed speech recognition; advanced front-end feature extraction algorithm; compression algorithms. 2002.

[6] 河原, 武田, 伊藤, 李, 鹿野, 山田: 連続音声認識コンソーシアムの活動報告及び最終版ソフトウェアの概要. 信学技報, SP2003-169, 2003

[7] 長友, 西村, 小松, 黒田, 李, 猿渡, 鹿野: “相補的バックオフを用いた言語モデル融合ツールの構築”, 情処学論, Vol.43, No.9, pp.2884-2893, 2002.

[8] 緒方, 後藤, 浅野: “話し言葉音声認識のための動的発音モデリング法の検討”, 音講論集, pp.203-204, 2004.

[9] http://www.btoday.net/broadcast/archives/009_nikkan/