US 20080109221A1

(19) **United States**

(12) **Patent Application Publication**

(10) **Pub. No.: US 2008/0109221 A1**

(43) **Pub. Date: May 8, 2008**

Doulton

(54) **MASS-SCALE, USER-INDEPENDENT, DEVICE-INDEPENDENT VOICE MESSAGING SYSTEM**

(75) Inventor: **Daniel Michael Doulton**, Buckinghamshire (GB)

Correspondence Address:
SYNNESTVEDT LECHNER & WOODBRIDGE LLP
P O BOX 592
112 NASSAU STREET
PRINCETON, NJ 08542-0592 (US)

(73) Assignee: **SPINVOX LIMITED**, Buckinghamshire (GB)

(21) Appl. No.: **11/931,097**

(22) Filed: **Oct. 31, 2007**

Related U.S. Application Data

(62) Division of application No. 11/673,746, filed on Feb. 12, 2007.

(30) **Foreign Application Priority Data**

Feb. 10, 2006 (GB) GB0602682.7
Jan. 9, 2007 (GB) GB0700376.6
Jan. 9, 2007 (GB) GB0700377.5

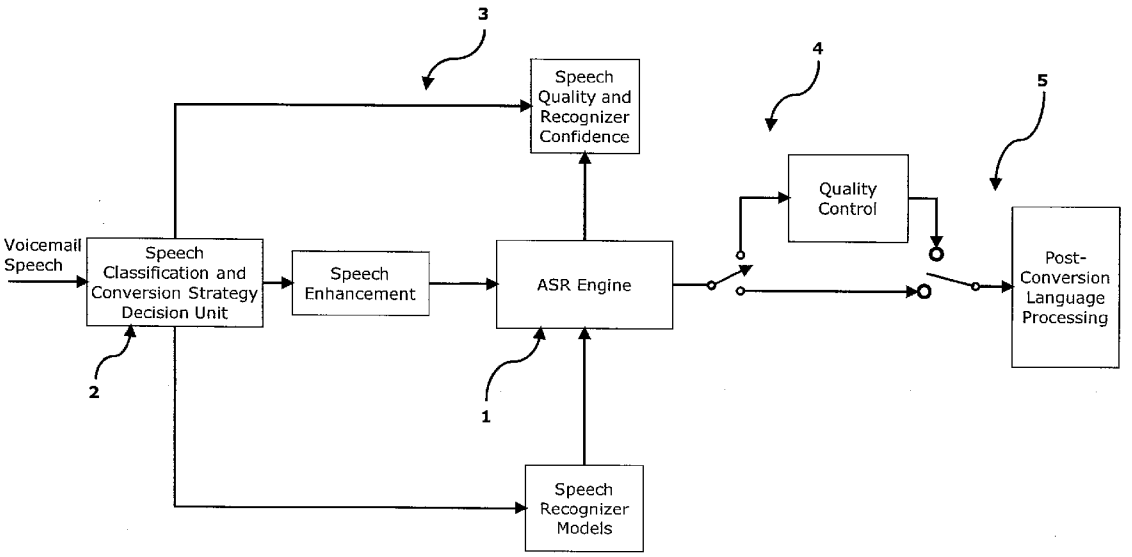
Publication Classification

(51) **Int. Cl.**
G10L 15/26 (2006.01)

(52) **U.S. Cl.** **704/235**

(57) **ABSTRACT**

A mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen is disclosed. The system comprises (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising 3 core sub-systems, namely (i) a pre-processing front end that determines an appropriate conversion strategy; (ii) one or more conversion resources; and (iii) a quality control sub-system.



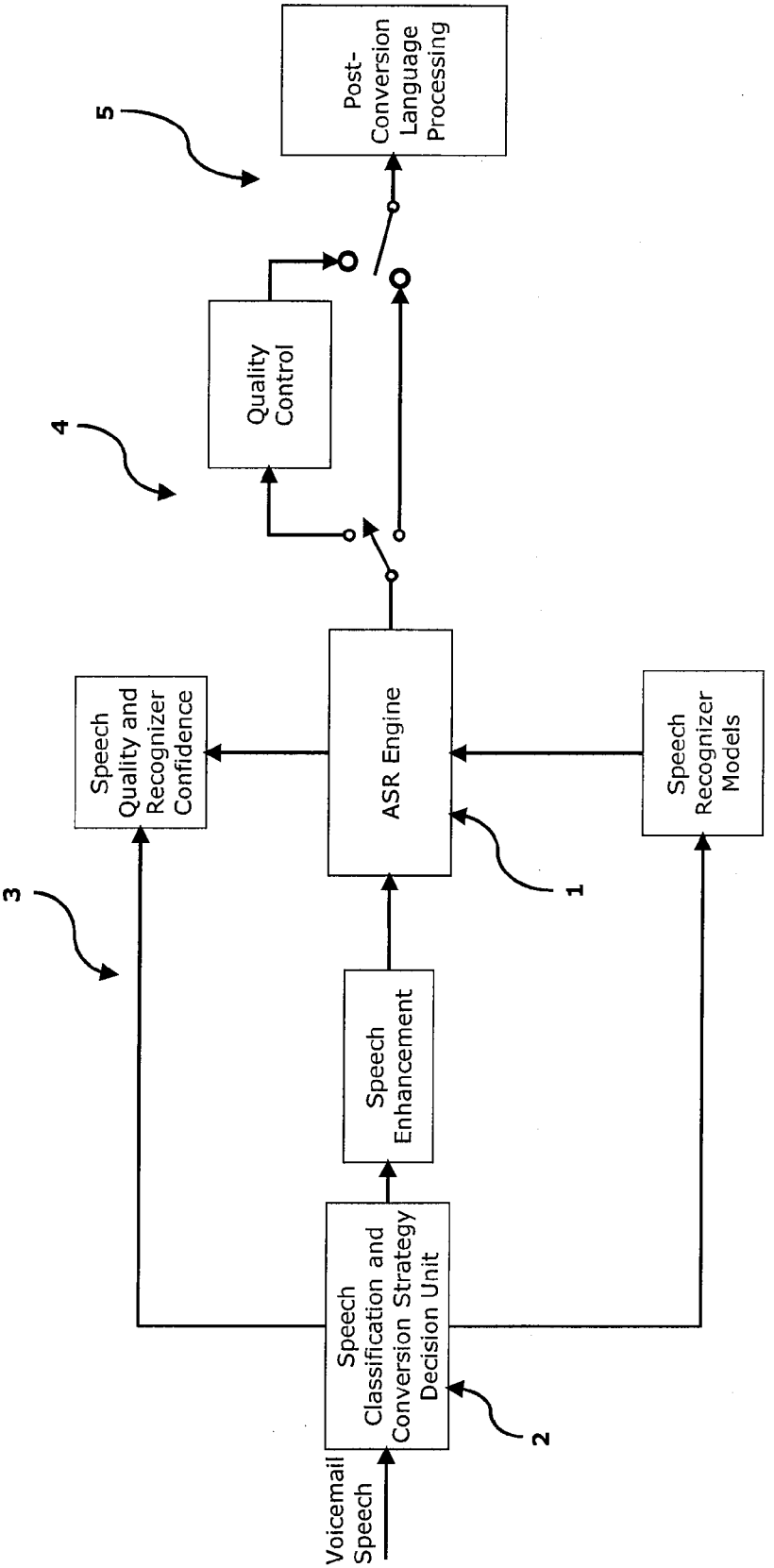

Post-Conversion Language Processing
5
Quality Control
4
Speech Quality and Recognizer Confidence
ASR Engine
Speech Recognizer Models
3
1
Speech Enhancement
Speech Classification and Conversion Strategy Decision Unit
2
Voicemail Speech

Figure 1

Figure 2

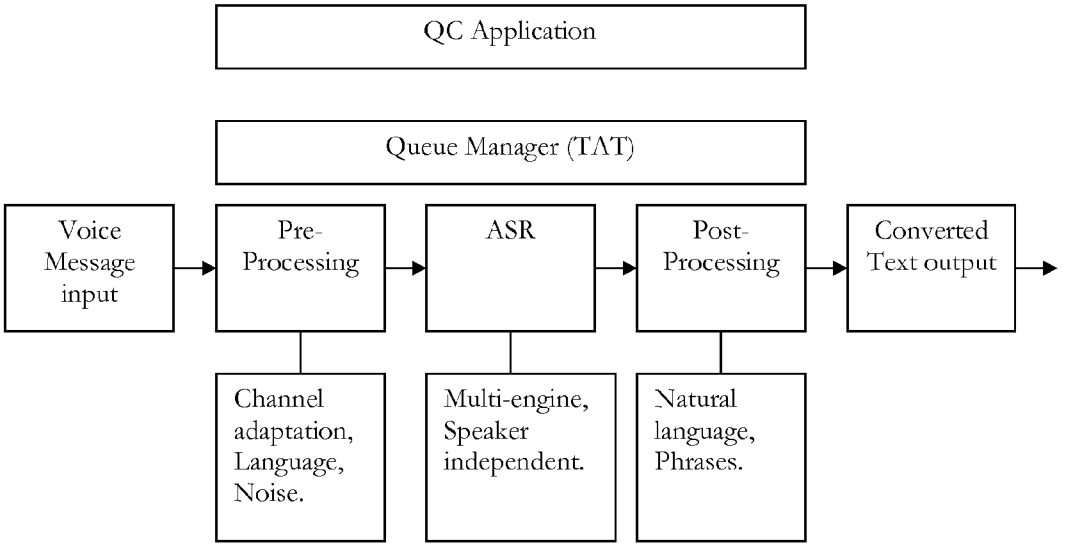

QC Application
Queue Manager (TAT)
Voice Message input
Pre-Processing
ASR
Post-Processing
Converted Text output
Channel adaptation, Language, Noise.
Multi-engine, Speaker independent.
Natural language, Phrases.

Figure 3

Correct: hi jonathan i will be in the stag and hounds at seven forty see you soon andy

Display: | hi john it's tam i will be into stagecoach after four to meet you soon amy

Input: <*accept_word*>

D: **hi** | john it's tam i will be into stagecoach after four to meet you soon amy

Input: 3 * <*accept_char*>

D: **hi jo** | hn it's tam i will be into stagecoach after four to meet you soon amy

Input: n

D: **hi jon** | athan i will be into stagecoach after four to meet you soon amy

Input: 4 * <*accept_word*> 3 * <*accept_char*>

D: **hi jonathan i will be in** | to stagecoach after four to meet you soon amy

Input: <*space*>

D: **hi jonathan i will be in** | the stadium from after four to meet you soon amy

Input: <*accept_word*> 4 * <*accept_char*>

D: **hi jonathan i will be in the sta** | dium from after four to meet you soon amy

Input: g

D: **hi jonathan i will be in the stag** | ecoach after four to meet you soon amy

Input: ' '

D: **hi jonathan i will be in the stag** | and hounds after four to meet you soon amy

Input: 3 * <*accept_word*> 2 * <*accept_char*>

D: **hi jonathan i will be in the stag and hounds a** | fter four to meet you soon amy

Input: t

D: **hi jonathan i will be in the stag and hounds at** | seven forty see you soon amy

Input: 6 * <*accept_word*> 2 * <*accept_char*>

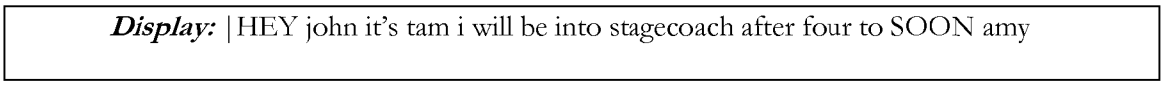
D: **hi jonathan i will be in the stag and hounds at seven forty see you soon a** | my

Input: n

D: **hi jonathan i will be in the stag and hounds at seven fourty see you soon a** | ndy

Input: <*accept_utterance*>

Figure 4


Display: |HEY john it's tam i will be into stagecoach after four to SOON amy

MASS-SCALE, USER-INDEPENDENT, DEVICE-INDEPENDENT VOICE MESSAGING SYSTEM

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application is a divisional of U.S. application Ser. No. 11/673,746, filed Feb. 12, 2007, which is based on and claims priority to Great Britain Application No. GB 0602682.7, filed Feb. 10, 2006, Great Britain Application No. GB 0700376.7, filed Jan. 9, 2007, and Great Britain Application No. GB0700377.5, filed Jan. 9, 2007, the contents of which are fully incorporated herein by reference.

BACKGROUND OF THE INVENTION

[0002] 1. Field of the Invention

[0003] This invention relates to a mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen. It is worthwhile initially looking at the challenges facing a mass-scale, user-independent, voice messaging system that can convert unstructured voice messages into text. First, 'mass-scale'—means that the system should be scalable to very large numbers, for example 500,000+ subscribers (typically these are subscribers to a mobile telephone operator) and still allow effective and fast processing times—a message is generally only useful if received within 2-5 minutes of being left. This is far more demanding than most ASR implementations. Second, 'user-independent': this means that there is absolutely no need for a user to train the system to recognise its voice or speech patterns (unlike conventional voice dictation systems). Third, 'device-independent': this means that the system is not tied to receiving inputs from a particular input device; some prior art systems require input from say a touch tone telephone. Fourth, 'unstructured': this means that messages have no pre-defined structure, unlike response to voice prompts. Fifth, 'voice messages': this is a very specific and quite narrow application field that raises different challenges to those faced by many conventional automated speech recognition (ASR) systems. For example, voice mail messages for a mobile telephone frequently includes hesitations, 'ers' and 'ums'. A conventional ASR approach would be to faithfully convert all utterances, even meaningless sounds. The mindset of accurate or verbose transcription characterises the approach of most workers in the ASR field. But it is in fact not appropriate at all for the voice messaging domain. In the voice messaging domain, the challenge is not accurate, verbose transcription at all, but instead capturing meaning in the most helpful manner for the intended recipient(s).

[0004] Only by successfully addressing all five of these requirements is it possible to have a successful implementation.

[0005] 2. Description of the Prior Art

[0006] Conversion from speech-to-text (STT) uses automatic speech recognition (ASR) and has, up until now, been applied mainly to dictation and command tasks. The use of ASR technology to convert voicemail to text is a novel application with several characteristics that are task specific. Reference may be made to WO 2004/095821 (the contents of which are incorporated by reference) which discloses a voice mail system from Spinvox Limited that allows voicemail for a mobile telephone to be converted to SMS text and sent to the mobile telephone. Managing voicemail in text form is an attractive option. It is usually faster to read than to listen to messages and, once in text form, voicemail messages can be stored and searched as easily as email or SMS text. In one implementation, subscribers to the SpinVox service divert their voicemail to a dedicated SpinVox phone number. Callers leave voicemail messages as usual for the subscriber. SpinVox then converts the messages from voice to text, aiming to capture the full meaning as well as stylistic and idiomatic elements of the message but without necessarily converting it word-for-word. Conversion is done with a significant level of human input. The text is then sent to the subscriber either as SMS text or email. As a result, subscribers can manage voicemail as easily and quickly as text and email messages and can use client applications to integrate their voicemail—now in searchable and archivable text form—with their other messages.

[0007] The problem with transcription systems that are significantly human based however is that they can be costly and difficult to scale to the mass-market—e.g. to a user base of 500,000+ or more. Consequently, it is impractical for major mobile or cell phone operators to offer them to their subscriber base because for the required fast response times it is just too expensive to have human operators listening to and transcribing the entirety of every message; the cost per message transcribed would be prohibitively high. The fundamental technical problem therefore is to design an IT-based system that enables the human transcriptionist to operate very efficiently.

[0008] WO 2004/095821 envisaged some degree of ASR front-end processing combined with human operators: in essence it was a hybrid system; the present invention develops this and defines specific tasks that the IT system can do that greatly increase the efficiency of the entire system.

[0009] Hybrid systems are known in other contexts, but the conventional approach to voice conversion is to eliminate the human element entirely; this is the mindset of those skilled in the ASR arts, especially the STT arts. We will therefore consider now some of the technical background to STT.

[0010] The core technology of speech-to-text (STT) is classification. Classification aims to determine to which 'class' some given data belongs. Maximum likelihood estimation (ILE), like many statistical tools, makes use of an underlying model of the data-generating process—be it the toss of a coin or human speech production system. The parameters of the underlying model are estimated so as to maximize the probability that the model generated the data. Classification decisions are then made by comparing features obtained from the test data with model parameters obtained from training data for each class. The test data is then classified as belonging to the class with the best match. The likelihood function describes how the probability of observing the data varies with the parameters of the model. The maximum likelihood can be found from the turning points in the likelihood function if the function and its derivatives are available or can be estimated. Methods for maximum likelihood estimation include simple gradient descent as well as faster Gauss-Newton methods. However, if the likelihood function and its derivatives are not available, algorithms based on the principles of Expectation-Maximization (EM) can be employed

which, starting from an initial estimate, converge to a local maximum of the likelihood function of the observed data.

[0011] In the case of STT, supervised classification is used in which the classes are defined by training data most commonly as triphone units, meaning a particular phoneme spoken in the context of the preceding and following phoneme. (Unsupervised classification, in which the classes are deduced by the classifier, can be thought of as clustering of the data.) Classification in STT is required not only to determine which triphone class each sound in the speech signal belongs to but, very importantly, what sequence of triphones is most likely. This is usually achieved by modelling speech with a hidden Markov model (HMM) which represents the way in which the features of speech vary with time. The parameters of the HMM can be found using the Baum-Welch algorithm which is a form of EM.

[0012] The classification task addressed by the SpinVox system can be stated in a simplified form as: "Of all the possible strings of text that could be used to represent the message, which string is the most likely given the recorded voicemail speech signal and the properties of language used in voicemail?" It is immediately clear that this is a classification problem of enormous dimension and complexity.

[0013] Automatic speech recognition (ASR) engines have been under development for more than twenty years in research laboratories around the world. In the recent past, the driving applications for continuous speech, wide vocabulary ASR have included dictation systems and call centre automation of which "Naturally Speaking" (Nuance) and "How May I Help You" (AT&T) are important examples. It has become clear that successful deployment of voice-based systems depends as heavily on system design as it does on ASR performance and, possibly because of this factor, ASR-based systems have not yet been taken up by the majority of IT and telecommunications users.

[0014] ASR engines have three main elements. 1. Feature extraction is performed on the input speech signal about every 20 ms to extract a representation of the speech that is compact and as free as possible of artefacts including phase distortion and handset variations. Mel-frequency cepstral coefficients are often chosen and it is known that linear transformations can be performed on the coefficients prior to recognition in order to improve their capability for discrimination between the various sounds of speech. 2. ASR engines employ a set of models, often based on triphone units, representing all the various speech sounds and their preceding and following transitions. The parameters of these models are learnt by the system prior to deployment using appropriate training examples of speech. The training procedure estimates the probability of occurrence of each sound, the probability of all possible transitions and a set of grammar rules that constrain the word sequence and sentence structure of the ASR output. 3. ASR engines use a pattern classifier to determine the most probable text given the input speech signal. Hidden Markov model classifiers are often preferred since they can classify a sequence of sounds independently of the rate of speaking and have a structure well suited to speech modelling.

[0015] An ASR engine outputs the most likely text in the sense that the match between the features of the input speech and the corresponding models is optimized. In addition, however, ASR must also take into account the likelihood of occurrence of the recognizer output text in the target language. As a simple example, "see you at the cinema at eight" is a much more likely text than "see you at the cinema add eight", although analysis of the speech waveform would more likely detect 'add' than 'at' in common English usage. The study of the statistics of occurrence of elements of language is referred to as language modelling. It is common in ASR to use both acoustic modelling, referring to analysis of the speech waveform, as well as language modelling to improve significantly the recognition performance.

[0016] The simplest language model is a unigram model which contains the frequency of occurrence of each word in the vocabulary. Such a model would be built by analysing extensive texts to estimate the likelihood of occurrence of each word. More sophisticated modelling employs n-gram models that contain the frequency of occurrence of strings of n elements in length. It is common to use $n=2$ (bigram) or $n=3$ (trigram). Such language models are substantially more computationally expensive but are able to capture language usage much more specifically than unigram models. For example, bigram word models are able to indicate a high likelihood that 'degrees' will be followed by 'centigrade' or 'Fahrenheit' and a low likelihood that it is followed by 'centipede' or 'foreigner'. Research on language modelling is underway worldwide. Issues include improvement of the intrinsic quality of the models, introduction of syntactic structural constraints into the models and the development of computationally efficient ways to adapt language models to different languages and accents.

[0017] The best wide vocabulary speaker independent continuous speech ASR systems claim recognition rates above 95%, meaning less than one word error in twenty. However, this error rate is much too high to win the user confidence necessary for large scale take up of the technology. Furthermore, ASR performance falls drastically when the speech contains noise or if the characteristics of the speech do not match well with the characteristics of the data used to train the recognizer models. Specialized or colloquial vocabulary is also not well recognized without additional training.

[0018] To build and deploy successful ASR-based voice systems clearly requires specific optimization of the technology to the application and added reliability and robustness obtained at the system level.

[0019] To date, no-one has fully explored the practical design requirements for a mass-scale, user-independent, hybrid voice messaging system that can convert unstructured voice messages into text. Key applications are for converting voicemail sent to a mobile telephone to text and email; other applications where a user wishes to speak a message instead of typing it out on a keyboard (of any format) are also possible, such as instant messaging, where a user speaks a response that it captured as part of an IM thread; speak-a-text, where a user speaks a message that he intends to be sent as a text message, whether as an originating communication, or a response to a voice message or a text or some other communication; speak-a-blog, where a user speaks the words he wishes to appear on a blog and those words are then converted to text and added to the blog. In fact, wherever there is a requirement, or potential benefit to be gained from, enabling a user to speak a message instead of having to directly input that message as text, and having that message converted to text and appear on screen, then mass-scale, user-independent, hybrid voice messaging systems of the kind described in the present specification may be used.

SUMMARY OF THE INVENTION

[0020] The invention is a mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen; the system comprising (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising:

[0021] 3 core sub-systems, namely (i) a pre-processing front end that determines an appropriate conversion strategy; (ii) one or more conversion resources; and (iii) a quality control sub-system.

[0022] Further aspects are given in Appendix III. The invention is a contribution to the field of designing a mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen. As explained earlier, this field presents many different challenges to the system designer compared to other areas in which ASR has in the past been deployed.

BRIEF DESCRIPTION OF THE DRAWINGS

[0023] The invention will be described with reference to the accompanying Figures, in which

[0024] FIGS. **1** and **2** are schematic views of a mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen, as defined by this invention.

[0025] FIGS. **3** and **4** are examples of how the system presents possible word and phrase choices to a human operator to accept or refine.

DETAILED DESCRIPTION

[0026] The SpinVox System Designers Faced Many Challenges:

[0027] Automatic Speech Recognition and Language Models

[0028] First and foremost, it was clear to the designers that established ASR technology on its own was not sufficient to provide reliable STT for voicemail (and other mass-scale, user-independent voice messaging applications). ASR relies on assumptions built from theoretical models of speech and language including, for example, language models containing word prior probabilities and grammar rules. Many, if not all, of these assumptions and rules are invalid in general for voicemail speech. Factors found in the voicemail STT application that are beyond the capabilities of standard ASR technology include:

[0029] voice quality is subject to environmental noise, handset and codec variation, network artefacts including noise and drop-outs;

[0030] users do not know they are talking to an ASR system and are comfortable leaving a message using natural and sometimes ill-structured language;

[0031] the language itself and accent used in the voicemail are not constrained or predictable;

[0032] vocabulary variations occur rapidly even within the same language so that, for example, the language statistics may vary because of major current affairs events.

[0033] IT Infrastructure

[0034] The design of the IT infrastructure to maintain availability and quality of the SpinVox service makes exacting demands on computational power, network and storage bandwidth and server availability. Loading on the SpinVox system is subject to unpredictable peaks as well as more predictable cyclic variations.

[0035] Unconvertible Messages

[0036] It is to be expected that a fraction of messages are unconvertible. These might be empty messages, such as 'slam-downs', messages in an unsupported language or unintentionally dialled calls.

[0037] Quality Assessment

[0038] The assessment of quality at each stage of the SpinVox system is in itself a challenge. Signal processing provides numerous analysis techniques that can be applied to the speech signal, ranging for straightforward SNR measurement to more sophisticated techniques including the explicit detection of common artefacts. However, direct measurements such as these are not significant in themselves but need to be assessed in terms of their impact on the later conversion process. Likewise, ASR confidence can be measured in terms of the output probability of alternative recognition hypotheses but, as before, it is important to measure the quality in terms of impact on the overall text conversion and the complexity of the quality control needed to reach it.

[0039] User Experience and Human Factors

[0040] The value of the system to customers is influenced heavily by the level of success with which human factors are accommodated by the design. Users will quickly lose confidence in the system if they receive garbled messages or find the system other than transparent and very simple to use.

[0041] The above challenges have been met in the SpinVox system design as follows:

[0042] System Design A simplified block diagram of the SpinVox system design in FIG. **1** shows the major functional units. At the core is the ASR engine **1**. SpinVox draws a clear distinction between ASR and a full STT conversion. ASR **1** is a subsystem which generates 'raw' text, given the input speech signal. This is a key element used for STT conversion, but is only one of several important subsystems that are needed to achieve reliable STT conversion. The front-end pre-processing subsystem **2** can perform broad classification of the speech signal which can be used to determine the conversion strategy in terms of the choice of a combination of ASR engine, model set and speech enhancement processing. The quality assessment subsystem **3** measures the quality of the input speech and the ASR confidence from which a quality control strategy can be determined. The quality control subsystem **4** operates on the ASR output. Its purpose is to generate semantically correct, meaningful and idiomatic text to represent the message within the constraints of the text format. Knowledge of context, including the caller ID, the recipient and caller-specific language models built up over time can be used to improve the quality of conversion sub-

stantially compared to the raw ASR output. The converted text is finally output from post conversion language processing sub-system **5** to an SMS text and email gateway.

[**0043**] Key Features The main features of the approach adopted by SpinVox are:

[**0044**] Meaningful Message Conversion

[**0045**] The text conversion captures the message—its meaning, style and idioms—but is not necessarily a word-for-word conversion of the voicemail.

[**0046**] Turn-around Time

[**0047**] The turn-around time for conversion of a message is guaranteed.

[**0048**] Reliability

[**0049**] The system can never send a garbled text message. Subscribers are alerted to unconvertible messages which can be listened to in the conventional way.

[**0050**] Standard Language

[**0051**] Messages are sent in standard language and are not 'textified'.

[**0052**] Wide Availability

[**0053**] The system operates entirely in the infrastructure and makes no requirements on the handset or network other than call divert.

[**0054**] Adaptive Operation

[**0055**] The system can optimize performance using embedded quality control strategies driven by knowledge built up over time from language modelling of voicemail in general as well as caller-specific language modelling. In addition, the system can choose from a number of possible speech-to-text conversion strategies based on the characteristics of the voicemail message. Voicemail message data and corresponding text conversions are continuously analysed so as to update and adapt the SpinVox STT system.

[**0056**] Quality Monitoring

[**0057**] The quality of speech-to-text conversion can be monitored at each stage and so quality control, whether by human or automatic agents, can be undertaken efficiently.

[**0058**] Language Processing

[**0059**] Post-conversion language processing **5** can be performed to enhance the quality of the converted message text, remove obvious redundancies and validate elements of the message such as commonly used salutation structures.

[**0060**] State-of-the-art ASR

[**0061**] Commercial ASR engines can be used so as to take competitive advantage of state-of-the-art ASR technology. Different ASR engines can be called on to handle different messages or even different parts of the same message (with the decision unit **2** decided on which engine to use). The human operators themselves could also be considered as an instance of an ASR engine, suitable for some tasks but not others.

[**0062**] Stable and Secure

[**0063**] The service is run on highly stable and secure Unix servers and can adapt to the demand for various languages as different time-zones experience peaks spread throughout each 24 hour period.

[**0064**] Quality Control SpinVox have developed a detailed understanding of users' expectations and desires for telephony-based messaging systems. They have identified zero-tolerance of users to nonsensical speech-to-text conversion, particularly where there is evidence that the errors have been introduced by a machine rather than by human error. Quality control of the converted text is therefore of key importance. Three alternative quality strategies can be used; decision unit **2** selects the optimal one. (i) Messages for which the ASR conversion confidence is sufficiently high can be checked automatically by quality assessment sub-system **3** for conformance to quality standards. (ii) Messages for which the ASR conversion confidence is not sufficiently high can be routed to a human agent **4** for checking and, if necessary, correction. (iii) Messages for which the ASR conversion confidence is very low are flagged as unconvertible and the user is informed of the receipt of an unconvertible message. Unconvertible messages can be listened to by the user, if they wish, using a single key-press. The outcome of these strategies is that the SpinVox system is designed so that a failure to convert is favoured over generating a conversion containing errors. User confidence in the system is therefore protected. SpinVox's statistics indicate that a substantial percentage of voicemails are successfully converted.

[**0065**] One of the important tools used by SpinVox for improving the quality of converted messages is knowledge of the language (common phrases, common greetings and sign-offs etc) used in voicemail messages. From accumulated data gathered over time, statistical language models specific to voicemail speech can be developed and then used to guide the STT conversion process. This greatly improves accuracy of conversion for non-standard language constructions.

[**0066**] The most apparent feature of SpinVox is that it supplies a service that many users did not realize they needed but soon find they cannot manage without. It is the first real-time system that provides speech-to-text conversation of voicemails. Its impact for network operators is increased network traffic, from improved call continuity, both for voice and data. The operational success of the SpinVox system has been achieved by taking a design approach that is driven by quality-of-service first, and technology second. The system design is based on a detailed understanding of customer expectations of the service and, even more importantly from an engineering perspective, the strengths and weaknesses of ASR technology. By exploiting the strengths of ASR and factoring out the weaknesses by stringent quality control, SpinVox is an effective deployment that meets the practical design requirements for a mass-scale, user-independent, hybrid unstructured voice messaging system.

[**0067**] SpinVox have demonstrated success in delivering a voice-processing-based service by focusing its conversion technology on a very specific target application—voicemail conversion. The indication is that system design targeted to a very well defined application is a more productive approach than the seemingly endless search for ever decreasing improvements in raw performance measures of, for example, ASR engines. This approach opens up the possibility of new

application areas into which SpinVox's technology components and system design know-how could be deployed.

[0068] SpinVox has developed significant know-how as system architects for voice-based applications with in-house expertise covering speech recognition, telecoms applications, cellular networks and human factors. The opportunities for growth and development in advanced messaging technologies are likely to point to enabling the integration of voice and text messaging, thereby facilitating search, management and archiving of voicemail with all the same advantages currently enjoyed by email and SMS text messaging, including operational simplicity and self-documentation. Such developments are paralleled by the convergence of voice and data in other telecommunications systems.

[0069] The Spinvox system is moving from what, from the outside, is a speaker independent problem towards a speaker dependent problem which is a huge insight into making speech work in telephony. Why? Because it is using the fact that calls, messaging and other communication is driven by community use—i.e. 80% of voicemails come from just 7-8 people. SMS just 5-6. IM just 2-3. Spinvox uses 'call-pair' history to do several things:

[0070] 1. build a profile of what a certain caller says every time they call—a speaker dependent speaker model—how the caller speaks (intonation, etc. . . .);

[0071] 2. build a language model of what that caller says to someone—a speaker dependent language model—what the caller says (words, grammar, phrases, etc. . . .);

[0072] 3. in 1 & 2 we are really building a language model of how A speaks to B. This is more refined than just how A speaks in general. It is atypical to the messaging type (i.e. how you speak in Voicemail) and it is also atypical to how you speak to B (e.g. the way a person speak a message to his mother is very different in intonation/grammar/phrases/accent/etc. . . . than when he speaks to his wife).

[0073] 4. Spinvox is building speaker-receiver pair models that go from general speaker/language independence to dependent without any user input or training;

[0074] 5. Spinvox has the ability to use both party's language to each other (e.g. how I call back and leave a message) to further refine relevant words (e.g. dictionary), grammar/phrases, etc. . . .

[0075] Further details on these aspects of the Spinvox Voice Message Conversion System are given in Appendix I below.

APPENDIX I

[0076] SpinVox—Voice Message Conversion System

[0077] The SpinVox Voice Message Conversion System (VMCS) focuses on one thing—conversion of spoken messages into meaningful text equivalent. In this, it is unique as are the methods and technologies advanced herein.

[0078] Concept

[0079] A novel method of converting voice-messages into text using multi-stage Automatic recognition techniques and human assisted Quality Control and Quality Assurance techniques and processes. The automated and human elements interact directly with each other to generate live/real-time feedback which is core to the system always being able to learn from live data to stay in-tune and deliver consistent quality. It is also designed to take advantage of the inherent limits of AI (ASR) and greatly improve accuracy by use of contextual bounds, human guidance, and language data live from the Internet.

[0080] Issue

[0081] Traditional approaches to speech conversion have been very much geared at the Recogniser level and creating high quality automatic speech recognition in laboratory conditions where inputs are highly controlled and guarantee a high level of accuracy.

[0082] The problem is that in the real world, speech recognition has many other elements to contend with:

[0083] Random speakers—anyone can use it

[0084] Noisy input—background noise and poor quality of speaker

[0085] Poor and variable transmission quality with tossy compression and bad mobile handset connections

[0086] Grammatically incorrect speech, slang, or highly localised expressions

[0087] Contextual sensitive grammar or implied meaning from a unique context between message creator and recipient

[0088] Context changes within a message—context boundaries—that invalidate the use of normal grammar rules

[0089] to cite a few and they all constantly vary with time, so the actual source input isn't a defined problem in time, but a constantly evolving one.

[0090] Solution

[0091] Key is to correctly define the problem: conversion of spoken messages into meaningful text equivalent.

[0092] This does not mean perfect, verbose transcription, rather the most important pieces of the message presented in easily understandable form. Accuracy measures are both quantitative and qualitative as the ultimate score is User Rated Accuracy where the SpinVox VMCS scores a consistent 97%.

[0093] There are two key parts:

[0094] Use a constant, live feed-back mechanism for the learning system, human driven

[0095] Use contextual information to better define each conversion problem

[0096] Using contextual information helps the system better estimate the likelihood of something being said in a message given the message's:

[0097] Type

[0098] Length

[0099] Time of day

[0100] Geography

[0101] Caller context—both caller and recipient (call-pair history)

[0102] Recent events

[0103] Etc. . . .

[0104] and known language structure most likely to occur in certain message types—Natural Language, described below.

[0105] Natural Language

[0106] When analysing voice messages and spoken text messages, regular patterns occur in what people say, how they say it and in what order—Natural Language. This clearly varies by message type or context, so for ordering Pizza, it would be different.

[0107] For example, in voicemail, how people greet can be well defined with 35 or so of the most common expressions—"Hi, it's me", "Hello, it's Daniel here", "Hiya. How you doing?", "Watcha mate", "Alright?", etc. . . . and likewise good-bye's can be well defined by common expressions—"Ok, bye", "Cheers", "Cheers mate", "Thanks now. Bye. Cheers", etc. . . .

[0108] Clearly, different parts of a spoken message have implicit meaning and therefore using this key we can improve recognition accuracy by using this context to select the most likely classification of what was actually said.

[0109] Building these into large statistically related models is what is defined as our Natural Language Model, one for each language, including dialects within any language.

[0110] Context Vectors

[0111] Natural language is often governed by its context, so when converting the body of a message, the context of what has been said can be used to better estimate what was actually said—e.g. a call to a booking line will more likely see expressions related to enquiring and some specific names of the company, product and price, versus a call to a home phone number where friendly expressions regarding greetings, 'how are you', 'call me back', etc. . . . are much more likely.

[0112] Within voice-messages, we can use context vectors to better estimate the likely content and natural language set that applies:

[0113] CLI (or any party identifier) is a very powerful context vector

[0114] Can tell from the number's geography likely language/dialect and regional real nouns most likely to be used

[0115] Can tell if number is a known commercial number and therefore message type better predicted—e.g. calls from 0870 no.s are commercial and therefore high chance this is a business message, whereas from 07 range is from a personal mobile, so time of day will drive message type more between business, personal, social or other.

[0116] Allows you to better get their number right if spoken inside the message

[0117] Is a key from which you can build history and known dictionary/grammar—e.g. always says 'dat's wicked man' in a street accent

[0118] Can build a speaker dependent recognition system—i.e. we can tune the ASR to you as a particular caller and get much higher recognition accuracy, your own vocabulary, grammar, phraseology, lexicon and general natural language

[0119] Call Pair History—deeper use of CLI (or any party identifier)

[0120] You can train the system far more accurately to a call pair's message history

[0121] You can train to A party's (caller) voice regardless of B party (recipient)

[0122] You can train to subject area and language A party uses with B party

[0123] You can train multiple A & B party relationships and drive the system to higher accuracy and speed

[0124] Time of day, day of week

[0125] Voicemail traffic rates, average message length and content type vary with time of day in each language market, from very business-like messages during peak hours (8 am-6 pm) to more personal (7-10 pm), to highly social (11 pm-1 am), to very functional (2 am-6 am). This also varies by day of the week so a Wed is the busiest day and contains highest level of business messages, but Sat & Sun have a very different profile of message type (largely personal, chatty messages) that need to be treated differently.

[0126] International numbers

[0127] By parsing the country code (e.g. 44, 33, 39, 52, 01) we can better determine language and dialect.

[0128] Available customer data

[0129] Customer name, address and possibly work place.

[0130] Implied Context Between A & B Party

[0131] Taking this on step further, there are many other very important clues that can help us better estimate the likely content of a message, particularly those that relate to who the two parties are, what the likely purpose of the message is, and where they're calling from or to.

[0132] In voicemail-to-text and spoken text, we know that having the caller's number

[0133] Allows us to better estimate any number left inside the message

[0134] Build history of known words, expressions, phrases, etc. . . . between the two parties

[0135] Likely language (e.g. call from +33 to +33 will most likely be in French, but +33 to +44 may have a 50% chance of being in French)

[0136] Names and their correct spellings

[0137] If you know the history of the A party's calls/messages, and their history of messages for the B party, you can build a speaker dependent profile and gain large improvements from your recogniser and its grammar.

[0138] Conversion Quality

[0139] In addressing this problem, defining the actual required outcome is essential as it makes a big difference in your approach to solving how to convert voice-messages (voicemails, spoken SMS, Instant messaging, etc. . . .) to text and how to optimally apply the conversion resource you have.

[0140] When someone leaves us a voice message, the purpose is a message, not a formal written piece of communication, so less accurate conversion will be tolerated as long as the meaning of the message is correctly conveyed.

[0141] In addition, there is asymmetry so that the message depositor isn't comparing what they said with the converted text. With the context of who called, the recipient is reading the converted output with the objective of finding out what the message is, so the requirement is excellent message extraction for conversion, not a verbose (word-for-word, utterance-for-utterance) conversion. In fact, conversely, a verbose conversion, unless well dictated, is often perceived as a low quality message as it contains lots of inelegant and unwanted pieces of spoken message language (e.g. uhmms, ahhs, repeats, spellings of words, etc. . . .)

[0142] Therefore, quality in this context is about extracting the important elements of a message—Intelligent Conversion.

[0143] At it's simplest, there are three key elements that provide the most meaning and hence are essential to achieving message quality:

[0144] 1. Who is it from—huge value in understanding meaning from this context

[0145] 2. What's the purpose of the message—e.g. call me back urgently, running late, change of plan/timings, call me on this no., just to say hi, etc. . . .

[0146] 3. Any specific facts, the most common being:

[0147] a. Names

[0148] b. Numbers, phone numbers

[0149] c. Time

[0150] d. Address

[0151] Other information in the message is largely there to support conveying these key elements and often helps provide better context for these key elements.

[0152] Varying Quality Sensitivity within Message

[0153] What's also very important to understand is that we need to recognise that each key part of any message has a different role in delivering the message and we can therefore attribute another dimension of quality to each that we should aim to achieve during conversion.

[0154] Messages can be broken down into

[0155] Greeting (top)

[0156] Message (body)

[0157] Good-bye (tail)

[0158] The percentage of messages that contain any body is clearly a function of deposited message length, so we know that short messages (e.g. sub 7 seconds) typically only contain a Greeting and a Goodbye. Above this, the probability of a meaningful message body grows exponentially. This fact also helps us better estimate the likely conversion strategy we should use.

Greeting & Good-Byes

[0159] How someone greets you can be classified into a some 50 commonly recognised salutations (e.g. Hi there, hey it's me, Hi, this is X calling from Y, Hello, I'm just calling to . . . , etc. . . .). Likewise, the 'good-bye' element of a message can be classified into a similar order of commonly recognises good-byes (e.g. Thanks very much, see ya, Ta, good bye, Cheers now, see you later, etc. . . .)

[0160] Two issues dictate our conversion quality requirement:

[0161] 1. Greetings and Good-byes are there for message protocol and often contain little of value for the main message, so our tolerance to low accuracy is high, provided that it makes sense.

[0162] 2. We can classify the vast majority of Greetings and Good-byes into some 50 commonly recognised categories each.

[0163] Therefore, the quality requirement during a Greeting, a salutation or Good-bye is far less than what's contained in the message body, normally the point of a message or a key fact—e.g. call me on 020 7965 2000

[0164] Message Body

[0165] The message body naturally has a higher quality requirement, but it likewise can often be found to contain regular patterns of natural language that relate to the context and we can therefore also apply a degree of classification to better help us attain the correct answer.

[0166] A good example is:

[0167] "Hi Dan, John here"—Top of message (or greeting)

[0168] "can you give me a call back on 0207965200 when you get this"—Body of message

[0169] "Thanks a lot mate. Cheers. Bye bye."—Tail of message (or good-bye)

[0170] In this case, the Body is a well structured piece of Voicemail Language that the SpinVox Conversion System has learnt. It can then break the message down and correctly assign the body outcome.

[0171] Elements in this case that apply are:

[0172] A & B party known

[0173] Phone number is John's CLI, or seen before in his calls to others

[0174] Message length—sub 10 seconds, so more likely to be common expression

[0175] Time of day—working hours—John normally doesn't leave detailed messages in working hours, just short, simple messages.

[0176] SpinVox Voice Message Conversion System

[0177] Having correctly stated our problem and identified some very important features of speech and how it relates to the text equivalent, the SpinVox system (see FIG. **2**) was designed to take full advantage of these:

[0178] Spin Vox Voice Message Conversion System

[0179] This diagram shows the three key stages that enable us to optimise our ability to correctly convert voice-messages (voicemail, spoken SMS, Instant messages, voice clips, etc. . . .) to text.

[0180] A key concept is that the system uses the term Agent for any conversion resource, whether machine/computer based or human.

[0181] Pre-Processing

[0182] This does two things:

[0183] 1. Optimises the quality of the audio for our conversion system by removing noise, cleaning up known defects, normalising volume/signal energy, removing silent/empty sections, etc. . . .

[0184] 2. Classifies the message type for optimally routing the message for conversion, or not.

[0185] Classifying the message type is done using a range of 'Detectors':

[0186] Language

[0187] e.g. English UK/US/Auz/NZ/S.Africa/Canadian and then to dialect types within it (e.g. within UK—S. East, Cockney, Birmingham, Glasgow, N. Ireland, etc. . . .)

[0188] Allows us to determine if we support the language

[0189] Allows to select which conversion route to use: QC/QA profile, TAT rules (SLA), which ASR stage strategy (engine(s)) to load and which post-processing strategy to apply

[0190] Methods:

[0191] Statistical Language Identification

[0192] Prior art:

[0193] several methods of automatic language identification known

[0194] SpinVox solution:

[0195] base decision on context: knowledge about registration, location and call history of caller and receiver

[0196] Signal Based Language Identification

[0197] Problem with Prior art:

[0198] high accuracy methods require large-vocab speech recognition or at least phone recognition, hence expensive to produce and run

[0199] requirement for reliable and fast method based purely on recordings (labeled with language but nothing else)

[0200] SpinVox solution:

[0201] 1. automatically cluster speech data for each language (vector quantization)

[0202] 2. combine cluster centres

[0203] 3. use statistical model of sequence of clusters for each language to find best match

[0204] 4. build model of relationship between score differences between models and expected accuracy

[0205] 5. combine several versions of 1-4 (based on varying training data, feature extraction methods etc) until desired accuracy is achieved

[0206] Noise—SNR detector

[0207] If the amount of noise in a message is above a certain threshold, then it becomes increasingly difficult to correctly detect the message signal and convert. More meaningful is if the ratio of the signal-to-noise drops below a certain level, then you've a high degree of confidence you won't be able to convert the message.

[0208] SpinVox users value the fact that when they receive a notice that the message was unconvertible, the source audio is so poor that over 87% of the time they call or text the person directly back and continue the 'conversation'.

[0209] Speech Quality Estimator

[0210] If the quality of someone's speech is likely too low for either the conversion system or agent to use. OR, content that a user ought to listen to themselves—e.g. someone singing them a happy birthday

[0211] SpinVox solution includes:

[0212] 1. find drop-outs (voice packets lost during transmission) based mainly on zero-crossing counts

[0213] 2. also estimate noise levels

[0214] 3. calculate overall measure of voice quality and use adaptive threshold to reject lowest quality messages

[0215] Hang-up ('slam-down') detector

[0216] Messages where someone called, but left no meaningful audio content. Typically short messages with background utterances.

[0217] Inadvertent Call detector

[0218] Typically a call from the redial button being pressed whilst in someone's pocket and leaving a long rumbling message with no meaningful audio content in it

[0219] Standard messages

[0220] Pre-recorded messages, most common in US, from an auto-dialling system, or service notices or calls.

[0221] Greet & Good-bye

[0222] If the message only contains these, then we can use a dedicated piece of ASR to correctly convert these messages

[0223] Message length & speech density

[0224] Length allows us to initially estimate the likelihood of message type—e.g. short call normally just a simple 'hi, it's X, call me back please' vs. long call which will contain something more complex to convert

[0225] Speech density will allow you to adjust your estimate of how likely a message length is to be a good indicator of type—e.g. low density and short message is likely to be just a simple 'hi, it's X, call me back please', but high density, short message will skew this

towards you needing a higher level of conversion resource as message complexity will be higher.

[0226] Clearly, pre-processing allows us in certain cases (e.g. slam-down, inadvertent call, foreign/unsupported language) to immediately route the message as classified and send the correct notification text to the recipient (e.g. 'this person called, but left no message'), saving any further use of precious conversion system resources.

[0227] Automatic Speech Recognition (ASR)

[0228] This is a dynamic process. The optimal use of conversion resources is determined at a message level.

[0229] We take input from both the Pre-Processing stage on message classification and from any context vectors and use these to choose the optimal conversion strategy. This means that this stage is using the best ASR technology for the particular task. The reason is that different types of ASR are highly suited to specific tasks (e.g. one is excellent for greetings, another for phone numbers, another for French addresses).

[0230] This stage is designed to use a range of conversion agents, whether ASR or humans, and only discerns between them based on how the conversion logic is configured at that time. As the system learns, this is adapted and different strategies, conversion resources and sequences can be employed.

[0231] This strategy isn't only applied at the whole message level, but can be applied within a message.

[0232] Top 'n' Tail

[0233] A strategy is to sub-divide the message's Greeting (top), Body and Goodbye (tail) sections, send them to different pieces of ASR that are optimal for that element of a message. Once they've completed, they're re-assembled as a single message.

[0234] Number Routing

[0235] Another strategy is to sub-divide out any clear elements where phone numbers, currencies or obvious use of numbers is spoken in the message. These sections are sent to a specific piece of ASR or Agent for optimal conversion, then reassembled with the rest of the converted messages.

[0236] Address Routing

[0237] Likewise, sub-dividing out elements where an address is being spoken can be sent to a specific piece of ASR or Agent, and to an address co-ordinate validator to ensure that all pieces of an address converted are real. E.g. if you can't detect the street name, but have a clear post-code you can complete the most likely street name. The accuracy of finding the street name is improved by reprocessing the address again, but with your estimated a priori street name refining your ASR classification variables to a much more limited set and seeing if there's a high match or not.

[0238] Real Noun Routing

[0239] Real nouns are renowned for making ASR unreliable. Again, but focusing on just this part and applying much more specialised, but computationally more expensive resource, you can much better estimate the real noun.

[0240] Post Processing

[0241] The ASR stage contains its own dictionaries and grammar, but this isn't sufficient to correctly convert the many complex ways we speak. ASR is very much geared at the word level conversion and very short sequences of words for estimating likelihood of word sequences and basic grammar (n-gram and trellis techniques). One problem is that mathematically speaking, as you widen the set of words you try to estimate the possible combinations of beyond 3 or 4, the permutations become so large that your ability to pick the right one diminishes faster than any gain made by widening the number of sequential words, so it's currently an unreliable strategy.

[0242] A very good method is to look at wider phrases or sentence structures that occur in Natural Language. Approaching the problem from the macro level, you can estimate better solutions for errors or words/sections where the ASR confidence is low.

[0243] However, this too has its weaknesses. As mentioned, human speech contains much noise, artefacts and because of the intimate relationship between A and B parties, is prone to large contextual boundaries. Some things make no sense to anyone other than to people who have a huge set of context in which to make much sense out of seemingly random phrases or unreliable sounding words.

[0244] For instance, "see you by the tube at Piccadilly opposite the Trocadero and mine's a skinny mocha when you get here" would make no sense to anyone unless they knew the possible meaning of 'tube', they'd been to London and knew that Piccadilly has a building called the 'Trocadero' very close by and understood that in Starbucks nearby they serve a drink called a 'mocha' and it's low fat, i.e. 'skinny'.

[0245] Real-World Corpi—Context Check

[0246] One solution is to look at a very large corpus of English words, real-nouns, phrases, sayings, regular expressions to check that your conversion might have contained these sequence of words.

[0247] The problem is that in normal speech, there are enormous possible combinations of these, and critically, this lacks any real-world context check. How do you know that the combinations of real nouns Piccadilly, Trocadero, mocha and skinny are valid, let alone good conversions of your source audio? The only absolute is a real-world check and unfortunately, by definition, we humans are the only ones at the moment able to qualify whether something has real-world validity or not—we do after all still program computers and databases they rely on.

[0248] With human level intelligence, you can most accurately check whether these seemingly unconnected items have any likely context in the real world. However, humans also lack in complete knowledge of everything which is why a significant percentage of Londoners wouldn't be comfortable knowing whether this phrase was likely or not given their knowledge of Piccadilly.

[0249] A solution is to use the planet's largest corpi of human knowledge. The billions of pages and databases created by human editors available on the Internet. A simple query as to whether the sentence, or any element of your conversion, is cited on the internet gives you a highly qualified real-world test of whether this is something humans have likely experienced and recorded and therefore might be real.

So in the above example, we find that Google, Yahoo!, MSN and other major search engines are able to give enough page hits with these in that we have a highly improved confidence that our conversion is indeed correct.

[0250] Further, using the Internet, we can most often find the correct spelling of phonetic approximations of words, real-nouns and place names that ASR attempts with new or unknown words if comes across. Currently, this is done through enormously time consuming and expensive manual programming of the ASR's dictionaries.

[0251] The other extremely valuable benefit of this solution is that the Internet is a live system that very accurately reflects current language, which is an evolving and dynamic subject and can vary with a single news headline, so you're not reliant on constantly updating your ASR dictionaries with a limited sub-set of natural language, but have access to probably the planet's most current and largest source of natural language.

EXAMPLE

[0252] SpinVox converts the following audio:

[0253] Message from English person—

[0254] Audio: "The cat sat on Sky when Ronaldo scored against Caca"

[0255] Converted text:

[0256] The cat sat on sky when Rownowdo/Ron Al Doh/Ronaldow/Ronahldo scored against Caka/Caca/Caker

[0257] Problems:

[0258] 'sat on sky' is grammatically incorrect—you can't sit on 'sky' in dictionary context

[0259] Rownowdo/Ron Al Doh/Ronaldow/Ronahldo are possible solutions for an unusual real noun

[0260] Caka/Caca/Caker are guesses of a highly unusual real noun Searches on Google for trouble elements of this phrase show:

[0261] Sky is a brand name—no. 1 rank for 'on Sky'. Therefore, it is a real noun for an object, so 'The cat sat on Sky' is possible grammatically and valid

[0262] The first name is most likely Ronaldo just from spell checks alone of all versions (Google's "Did you mean: Ronaldo?")

[0263] Ronaldo is highly correlated with 'Ronaldo scored' as he's a very famous football player and the search returns a large number of exact matches for this phrase

[0264] The second name is most likely Caca, because Caca has most hits for 'scored against Caca'.

[0265] We further our confidence by searching for 'football Caca'—football being derived from the context of 'Ronaldo scored'- and we get a large number of highly correlated search results. Given 'Ronaldo scored' has already returned a large number of successful searches, we are more confident that 'Caca' is the most likely fit.

[0266] Further, the real-world nature of data indexing of Google means that terms being used today, current terms, get higher rankings than less current terms, which is essential in getting speech recognition to work for current language and context.

[0267] Queue Manager

[0268] The Queue Manager is responsible for:

[0269] Determining what should happen to a voice message at each stage—conversion strategy

[0270] Managing the decision of each Automated stage when it requires human assistance

[0271] If at any stage of the automated conversion confidence intervals or other measures suggest any part of a message won't be good enough, then the Queue Manager directs this to the correct human agent for assistance.

[0272] Guaranteeing our Service Level Agreement with any customer by ensuring we convert any message within an agreed time—Turn Around Time (TAT)

[0273] typically TAT is an average of 3 mins, 95% within 10 mins, 98% within 15 mins.

[0274] Making decisions by calculating trade-offs between conversion time and quality. This is a function of what the SLA allows, particularly to deal with unexpected traffic or abnormal language use spikes and performance to-date.

[0275] This is achieved by use of large state-machines that for any given language queue can decide how best to process the messages through the system. It interacts with all parts and is the operational core of the SpinVox VMCS.

[0276] Quality Control Application

[0277] Appendix II contains a fuller description of this Lattice method as used within the SpinVox Quality Control Application.

[0278] As shown in the Voice Message Conversion System (VMCS) FIG. **2** diagram, human agents interact with messages at various stages. They do this using the Quality Control Application.

[0279] They also use a variant of this tool to randomly inspect messages to ensure the system is correctly converting messages as one of the problems with AI is that it is unable to be sure that it really is accurate.

[0280] A key inventive step is the use of humans to 'guide' the conversion of each message. This relies on the SpinVox VMCS databases, which contain a large corpus of possible matches, ASR and a human inputting a few words to create predictive typing solutions. In its extreme case, no human is required and the conversion is fully automatic.

[0281] Issue

[0282] ASR is only good at word level matches. To convert a meaningful message, phrases, sentences and grammar for spoken messaging is necessary. ASR will produce a statistical confidence measure for each word level match, and phrase where available. It is unable to use context or natural language rules to complete the process of a meaningful and correct conversion.

[0283] What automated systems are good at is spelling and base grammar—consistency.

[0284] What humans are good at are meaning, context, natural language, spoken grammar, dealing with ambiguous input and making sense of it. Humans tend to be inconsistent with spelling, grammar and speed.

[0285] Business Issue

[0286] Using humans costs money, so anything that can be done to use them for only the things that matter, and hence of economic value, is essential.

[0287] SpinVox VMCS uses the concept of Agent Conversion Ratio (ACR)—the ratio of the time it takes and agent to actually process a message to the length of the spoken message. Anything that reduces ACR and improves message conversion quality is a business driver as a 1% reduction in ACR leads to at least a 1% improvement in gross margin. In fact, the sensitivity is even higher as not only is direct cost of goods sold reduced, but management overhead and operational availability of the service and scalability all benefit from fewer humans required.

[0288] Solution

[0289] Lattice method: use human agents to guide the system to pick the correct set of words, phrases, sentences, messages from a pre-determined list of most likely options.

[0290] The SpinVox VMCS databases hold a rich history of message data as large statistical models (dictionaries and grammar with context vectors that relate them) that can be drawn upon in two key ways:

[0291] Lattice Method

[0292] i. The VMCS language model uses context (e.g. call pair history, language, time of day, etc. . . . —see Context Vectors) to pick the most likely conversion (the Proposed Conversion) to show to the agent.

[0293] ii. As the message plays back, the agent selects either a letter to choose an alternative (can be just the first few letters of the correct word), or hits 'accept' to accept the proposed section of text and move on to the next section.

[0294] iii. As the agent types changes, the system uses this as both input to pick the new most likely conversion and as feedback (learning) so that the next time it is more likely to get the right match first time.

[0295] iv. What would normally require an agent to type a full message worth of characters (e.g. 250), only takes a few key-strokes to complete and in real-time or faster.

[0296] v. The agent output is now constrained to correct spelling, grammar and phraseology, or rules about these that control quality and better message meaning.

[0297] This can be presented to the agent in two key ways:

[0298] 1. ASR Assisted Proposed Conversion

[0299] In this case, ASR is first used to better predict which text should be the Proposed Conversion for the agent. It uses what's actually in the audio of the spoken message to reduce the set of possible conversion options to a minimum, thus improving accuracy and agent speed.

[0300] a. ASR can be used for the initial proposed conversion

[0301] b. ASR can then be continuously used as the agent inputs selections to further refine the remaining sections of the proposed conversion

[0302] Prior art: Humans correcting transcriptions with choice of word alternatives

[0303] Problem with prior art:

[0304] Corrections still time consuming

[0305] ASR engine could have made better decision (later in utterance) if the user correction had been known during decoding

[0306] 2. Full Predictive Text Typing

[0307] Just like 1. above, but where no ASR is used to select the Proposed Conversion shown to an agent. This is different to standard predictive text editors as it relies on specific history (VMCS language models and use of context vectors—e.g. call pair history) and works at phrase level and above.

[0308] Prior art: predict most frequent word (list of alternatives) given partial human input

[0309] Problem with prior art:

[0310] Most frequent word very often not the one the user wants

[0311] Predictions just for one word

[0312] In either case, the SpinVox VMCS language models are trained purely by humans, or by a combination of ASR and humans.

[0313] In the extreme case, the system is fully trained and is able to always pick the right Proposed Conversion first time and only require human assistance for Quality Assurance to randomly sample and check the VMCS is correctly self-governing.

APPENDIX II

Lattice Method

[0314] Assorted Observations and Assumptions

[0315] 1. Given the large vocabulary and the varying audio quality, it seems impossible to achieve high enough speech recognition accuracy for fully automatic conversion for more than a tiny fraction of utterances. Reliably detecting this fraction, i.e. deciding that no human check is needed is a very interesting longer-term research problem but probably not a realistic option in the short term.

[0316] 2. While a good operator has a target ACR of 3-4, the average is more like 6-8.

[0317] 3. Correcting an utterance that is already 90% correct takes about 1.2. (Source: SpinVox Operational Research 2005)

[0318] 4. 75% of correction time is spent on finding and selecting errors (Wald et al).

[0319] 5. Word selection lists (alternatives) reduce listening time (Burke 2006).

[0320] 6. Errors tend to cluster (Burke 2006).

[0321] 7. Double speed playback keeps intelligibility and users appear to prefer it after short training (Arons 97).

[0322] 8. Removing pauses and 50% faster playback give real-time factor ⅓ (Arons 97).

[0323] 9. According to Bain et al 2005, normal typing has ACR 6.3 which equals ACR for editing ASR output with 70% accuracy. Shadow transcription is mentioned as "viable" for live subtitling.

[0324] Approach

[0325] The main aim has to be reducing the agent conversion ratio (ACR) by using speech technology to support the agent. This can be achieved in a number of ways:

[0326] 1. Let the agent make the decisions we can't afford to get wrong, i.e. the overall meaning of the message or individual phrases. The machine can fill in the details.

[0327] 2. Offer predictions while the agent types/edits the utterance. This might not just save typing time but also helps avoiding spelling mistakes.

[0328] 3. Provide (simplified) capitalization and punctuation automatically so that the agent doesn't need to deal with these issues.

[0329] Call Handling Steps

[0330] 1. Agent listens to message at high speed (e.g. ½ real time).

[0331] 2. Agent presses button to select category (e.g. "please_call_back", "just_calling_back", . . . "general").

[0332] 3. In some cases, the utterance is immediately accepted. This will happen if the message follows a simple pattern defined for the message category, the speech quality was good and there are no important but easily confusable parts in the message (e.g. times).

[0333] 4. System proposes converted string, agent edits while system continuously (and instanty) updates proposed utterance using predictions based on speech recognition results.

[0334] 5. Agent presses key to accept utterance as soon as the displayed utterance is correct.

[0335] An example for call handling step 4 is shown in FIG. **3**.

[0336] In this example, the agent would need 35 key strokes to edit an utterance with 17 words and 78 characters:

[0337] 15*<accept_word> (e.g. tab)

[0338] 14*<accept_char> (e.g. right arrow)

[0339] 6*normal input

[0340] 1*<accept_utterance> (e.g. Enter)

[0341] Most of them should be very quick because the same key has to be pressed a few times. Only 6 of them require selecting a normal key.

[0342] Note that only 6 of the 17 words (350%) were correct in the utterance originally proposed by the system.

[0343] Implementation

[0344] Processing Steps

[0345] 1. A Speech Recognition engine (e.g. HTK) converts the utterance speech file into a lattice (i.e. word hypothesis graph—a directed, a-cyclic graph representing a huge number of possible word sequences).

[0346] 2. The lattice is re-scored to take into account phone number pair) specific information (e.g. names, frequent phrases in earlier calls etc).

[0347] 3. The lattice is augmented to enable very fast search during the editing phase (e.g. the most likely path to the end of the utterance is computed for each node and the arc starting this path stored, character-sub trees are added to each node representing decision points). "Families", i.e. several arcs differing only in their start and end times are combined within certain limits.

[0348] 4. When the agent selects a specific category (step 2 in "Call handling steps"), a corresponding grammar and language model are selected for parsing and dynamic re-scoring. When the category is "general", an unrestricted "grammar" is used.

[0349] 5. The highest scoring path through the lattice matching the selected category grammar (if appropriate) is selected.

[0350] 6. The result found in this way will be accepted immediately if:

[0351] a. The category is not "general".

[0352] b. The score difference to the highest scoring unconstrained path is within a given range. This range can be used as a parameter to dynamically control the tradeoff between speed and accuracy.

[0353] c. According to the grammar used to find the path, the utterance does not contain crucial parts that are easily confusable (e.g. times).

[0354] 7. When the user accepts words or characters, the system moves along the selected path through the lattice.

[0355] 8. As characters and words are accepted or typed, their color or font changes.

[0356] 9. When the agent types something, the system selects the highest scoring path (again taking into account the current grammar and possibly other, e.g. statistical information) that starts with the character(s) typed. This new path is then displayed.

[0357] 10. When an agent types a word not found in the lattice, it is automatically spell-checked and correction is offered if appropriate.

[0358] 11. After the agent presses <accept_uterance>, the text is processed to add capitalization and punctuation, correct spelling mistakes, replace number words by digits etc. This uses a robust probabilistic parser using grammars semi-automatically derived from the training data.

[0359] Audio Playback

[0360] The nodes in the lattice contain timing information and hence the system can keep track of the part of the message the agent is editing. The agent can configure how many seconds the system will play ahead. If the agent hesitates, the system plays back the utterance from the word before the current node.

Refinement Options

[0361] Mark Important and Unimportant Parts

[0362] i. Depending on the relevant category and grammar, specific parts of the displayed utterance text that are deemed crucial are highlighted while particularly unimportant parts (e.g. greeting phrases) are shaded out.

[0363] Use Phrase Classes for Unimportant Parts

[0364] Parts of the message are displayed as phrase classes instead of individual words. The agent only needs to confirm the class while the choice of the individual phrase is left to the ASR engine because a mistake in this area is considered unimportant. For instance, the class "HEY" could stand for "hi, hay, hey, hallo, hello" and the earlier example could be displayed as shown in FIG. **4**. In this version, the "<accept_word>" key applied to a phrase would accept the whole phrase. Typing a character changes back to word mode, i.e. the phrase class marker is replaced by individual words.

[0365] Limit Prediction Display

[0366] Displaying the wrong predictions might actually confuse the agent and it might be worth displaying only those (partial ones) the system is relatively certain about.

[0367] Alternatively, the relative confidence in various predictions could be color-coded in some way (cf. "confidence shading"), e.g. uncertain ones (usually further away from the cursor) are printed in very light gray while more reliable ones are shown darker and bolder.

[0368] Utterance Segmentation

[0369] Longer silence periods are detected and used to break the message up into segments. The user interface reflects the segmentation and an extra key is assigned to "<accept_segment>". This enables confirming larger phrases with one key press and also resynchronization if the agent types a word not extending the current path through the lattice.

[0370] Keep Cursor on Left Side of Screen

[0371] Have a big area in the middle of the screen that shows the current phrase in large letters. As editing goes on, move the text (keep the cursor in same position). Show only a few words left of the cursor. As words drop out of the middle area, they move to the top area (smaller font, gray). More phrases are displayed below, again small and gray.

[0372] Show Phrase Alternatives

[0373] Always or after key press, show alternative phrase completions like drop down menu right of cursor and allow selection with arrow keys. This means the agent doesn't need to think about the first characters of the correct word which should help for difficult words.

[0374] Move Cursor with Speech

[0375] As the message is played back, the word spoken is highlighted automatically and the cursor is moved to the start of the word.

[0376] Play Highlighted Region

[0377] The agent can select a region (e.g. left and right mouse key) and the system keeps playing the segment between the markers until the agent moves on.

[0378] Shadow Transcription for Individual Words or Phrases

[0379] Words are highlighted as they are played to the agent and in addition to typing, the agent can simply say the word to replace the currently highlighted one (and the rest of the phrase). The system dynamically builds a grammar from the alternative candidates in the lattice (words and phrases) and uses ASR to select the correct one. This is a technically difficult option because ASR needs to be used from within the QC app and the appropriate speaker-dependent models need to be trained and selected at run time.

[0380] Accuracy Considerations

[0381] Most Likely Result

[0382] The accuracy of the highest scoring result after the speech recognition step is expected to be rather low (e.g. 25%). So the result displayed initially will only rarely be correct. IBM reports a word error rate of 28% for voice mail in Padmanabhan et al 2002.

[0383] When an utterance category can be identified (guessing: 20% of cases), the chance of getting a correct overall result should be reasonably high (say 70%) if the "phrase class" approach is used, i.e. if errors in the exact phrases used for top and tail are accepted and either there are no difficult parts in the message or they can be verified using other information (phone owners, previous calls). A rough guess would be that overall about 10% of utterances could be handled with just one key-press (the one needed for category selection).

[0384] Error Correction

[0385] It has been observed that speech recognition errors tend to occur in clusters, e.g. the average number of subsequent words containing an error is about 2 (TODO: find reference). This is usually due to:

[0386] segmentation errors—the first incorrect word is shorter or longer than the correct one and hence the next word must be wrong as well

[0387] the influence of the language model

[0388] possibly co-articulation modeling

[0389] This observation motivates the expectation that a correction of one word during the editing process will typically correct more than one mistake in the utterance hypothesis.

[0390] Very roughly speaking, typing one character limits the number of contenders for the next word by a factor of 1/26. Two characters limit it to 1/676 and should almost certainly exclude all higher scoring incorrect ones. This motivates another prediction: one ASR error should in the average require not more than one keystroke to correct.

[0391] Best Path Through Lattice

[0392] A very important factor for the success of the system is the percentage of lattices containing the correct path even if it has a comparatively low score. If the correct path is not in the lattice, the system will at some point not be able to follow a path through the lattice and hence it will be difficult to produce new predictions. The system might need to wait for the agent to type two or three words before finding appropriate points in the remaining lattice again to create further predictions.

[0393] The size of the lattice and hence the chance of getting the correct utterance can be controlled by parameters (number of tokens and pruning) and in theory the whole search space could be included. This would produce huge lattices, however, that could not be transmitted to the client within an acceptable time frame. Furthermore, we have to deal with the occasional occurrence of previously unseen words that would consequently not be in the vocabulary. After a few months of operation (and hence data collection) a rate of about 95% seems achievable.

[0394] If the "Utterance segmentation" version described above is used, the segments would provide easy points for re-starting prediction.

[0395] Linguistic Post-Processing

[0396] It might be worth defining a simplified "SpinVox message" syntax for each language. SMSs are generally not expected to contain full, proper sentences and rather than attempting to add a lot of punctuation (and often getting it wrong), it might be worth to use it rarely but consistently.

[0397] Capitalization

[0398] This is comparatively easy in English but more difficult in other languages (e.g. German).

[0399] Expected Benefits

[0400] 1. While converting or editing, the system keeps track of where in the utterance the agent currently is and hence audio playback can be controlled better.

[0401] 2. For a certain proportion of utterances, where the agent only needs to determine the category, the ACR could be less than one (theoretically ⅓ with fast playback and silence removal).

[0402] 3. A significant number of messages, for which the ASR performance is high, will require only a quick check and very few keystrokes to make corrections, giving an ACR of about 2.

[0403] 4. Most messages will still need significant editing. To what extent these cases will benefit from the predictions still has to be determined.

[0404] 5. Handling capitalization and punctuation automatically should reduce ACR by a small percentage and also improve consistency.

[0405] Questions/Issues

[0406] 1. When does editing with ASR-controlled predictions become more time consuming then simply typing? To take advantage of the predictions, the agent needs to read them. If the next few words are predicted correctly, simply accepting them should be faster than typing them but if the next word is wrong, the additional time required to check it is simply wasted. On the other hand, the agent needs to listen anyway and might as well use the time for checking the predictions.

[0407] Combining Prediction Methods

[0408] It seems promising to use statistical predictions as a back-up for ASR-based prediction.

[0409] Since the statistical prediction model is static (not call dependent) and hence doesn't need to be transmitted to the QC application with every message, it can be rather comprehensive. Lattices have to be transmitted for each message and hence will have to be kept within certain size limits and are likely to miss some of the hypothesis needed.

[0410] Both the statistical and the ASR-based prediction models would be represented as graphs and the task of combining the predictions would involve traversing both graphs separately and then either choosing the more reliable prediction or combining them according to some interpolation formula.

[0411] This method could be extended to more prediction model graphs, for instance based on call pairs.

[0412] Statistical Predictions

[0413] These predictions are based on n-gram language models. These models store conditional word sequence probabilities. For instance, a 4-gram model might store the probability of the word "to" following the three word context "I am going". These models can be very big and efficient ways of storing them are required that also enable fast generation of predictions.

[0414] Implementation

[0415] N-gram models are typically stored in a graph structure where each node represents a context (already transcribed words) and each outgoing link is annotated with a word and the corresponding conditional probability.

[0416] Since there will always be words that were never (or very rarely) encountered after a certain context but that are still required at run time, the model needs a way of dealing with previously unseen words in a given context. This is achieved by "backing-off" to the corresponding shorter context. In our example, if "to" had not been observed after "I am going", the model would look for "to" after "am going". If it wasn't observed there either, it would look at the context node for "going" and finally at the empty context node where all the words in the vocabulary are represented. This "backing off" is implemented by adding a special link to each context node that points to the node with the corresponding shorter context and is annotated with a "back-off penalty" that can be interpreted as the (logarithm of the probability mass not distributed over all the other links outgoing from the node.

[0417] The overall log probability of "to" after "I am going" could for instance be calculated as back_off("I am going")+back_off("am going")+link_prob("to"@context node "going").

[0418] Word Graph Expansion

[0419] It would be computationally expensive to search for the most likely word (link) starting with a given character sequence every time the user presses a key. One way to speed this up relies on expanding the word graph into a character graph where the outgoing links at each node are sorted by

decreasing likelihood. Note that the maximum number of outgoing links at each node is the number of characters in the language plus two for the back-off link and the word end link. Hence searching through this list would require at most about 100 character comparisons for English with the expected cost less than about 50 comparisons taking into account that the most likely words will be tried first.

[0420] This ignores the cost of searching for words not found at the current context node. When this is required, it might be best to accept that predictions can't be generated very quickly and to use the normal back-off link to search at the back-ff context nodes. The alternative of storing back-off links at each character node would require too much memory.

[0421] The expansion from the word graph to the character graph can be implemented in the following way:

[0422] 1. For each (word-level) context node:

[0423] sort all outgoing links by their probability (decreasing).

[0424] For each link (in order):

[0425] set pointer to current (word) node

[0426] for each character:

[0427] if there is already a link annotated with this character, set pointer to the node the link points to

[0428] else: add new link to the node the pointer points to and create a new node as destination for the link. Set pointer to this new node.

[0429] Add new link to pointer target, pointing to the destination of the current word link.

[0430] After the expansion, all word links (including their probabilities) can be deleted, except for the back-off links. Note that this will allow to always find the most likely phrase prediction but not the list of less likely ones. If this is required (later), the sequence in which the character expansion was performed would have to be stored in some way.

[0431] Prediction

[0432] Taking a word node id, a character node id, the current word sub-string and a character as input, the "prediction" method would:

[0433] 1. Goto character node[character_node-id]

[0434] 2. Find link annotated with the input character (use linear search in likelihood sorted links)

[0435] 3. If found, follow the link and starting from its target node, follow the first link leaving each node until some stop condition is reached. At each transition, add the character found at the link to the result string. Return the node id of the initial target node and the result string.

[0436] 4. Else: use back-off link from word_nodes [word_node_id] and the current word string to find predictions at the back-off node(s). This is not expected to find predictions at real-time if the user types quickly.

APPENDIX III

[0437] Core Concepts

[0438] The following concepts are covered. Each Core Concept A-I can be combined with any other core concept in an implementation.

[0439] The following text also describes various sub-systems that, inter alia, implement features of the Core Concepts. These sub-systems need not be separate from one another; for example, one sub-system may be part of another of the sub-systems. Nor do the sub-systems have to be discrete in any other way; code implementing functions of one sub-system can form part of the same software program as code implementing functions of another sub-system.

Core Concept A

[0440] A mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen; the system comprising (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising:

[0441] 3 core sub-systems, namely (i) a pre-processing front end that determines an appropriate conversion strategy; (ii) one or more conversion resources; and (iii) a quality control sub-system.

[0442] Other features:

[0443] The conversion resources include one or more of the following: one or more ASR engines; signal processing resources; the human operators.

[0444] The signal processing resources optimise the quality of the audio for conversion by performing one or more of the following functions: removing noise, cleaning up known defects, normalising volume/signal energy, removing silent/empty sections.

[0445] Human operators perform random quality assurance testing on converted messages and provide feedback to the pre-processing front-end and/or the conversion resources.

[0446] Core Concept B

[0447] Context Vectors

[0448] A mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen; the system comprising (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising:

[0449] a computer implemented context sub-system adapted to use information about the context of a message or a part of a message to improve the conversion accuracy.

[0450] Other features:

[0451] the context information is used to limit the vocabulary used in any ASR engine or refine search or matching processes used by the ASR engine.

[0452] the context information is used to select a particular conversion resource or combination of conversion resources, such as a particular ASR engine.

[0453] context information includes one or more of caller ID, recipient ID, whether the caller or recipient is a business or other type of classifiable entity or not; caller-specific language; call-pair history; time of call; day of call; geo-reference or other location data of the caller or callee; PIM data (personal information management data, including address book, diary) of the caller or callee; the message type, including whether the message is a voice mail, a spoken text, an instant message, a blog entry, an e-mail, a memo, or a note; message length; information discoverable using an online corpus of knowledge; presence data; speech density of the message; speech quality of the message.

[0454] the context sub-system includes a recogniser confidence sub-system that determines automatically the confidence level associated with a conversion of a specific message, or part of a message, using the context information.

[0455] the context sub-system includes or is connected to a recogniser confidence sub-system that determines automatically the confidence level associated with a conversion of a specific message, or part of a message, using the output of one or more ASR engines.

[0456] the recogniser confidence sub-system can dynamically weight how it uses the output of different ASR engines depending on their likely effectiveness or accuracy.

[0457] knowledge of the context of a message is extracted by one sub-system and fed-forward to a downstream sub-system that uses that context information to improve conversion performance

[0458] a downstream sub-system is a quality monitoring and/or assurance and control sub-system.

[0459] Core Concept C

[0460] Call-Pair History

[0461] A mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen; the system comprising (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising:

[0462] a computer implemented call-pair sub-system adapted to use call-pair history information to improve conversion accuracy.

[0463] Other features:

[0464] the call-pair history enables the system to be user-independent but to acquire over time, without explicit user training, user-dependent data that enables conversion performance to be improved.

[0465] the call-pair history is associated with a pair of numbers, including numbers associated with mobile telephones, fixed telephones, IP addresses, e-mail addresses, or unique addresses provided by a network.

[0466] The call-pair history includes information relating to one or more of: likely language or dialect being used; country called from or called to; time zones; time of call; day of call; specific phrases used; caller-specific language; intonation; PIM data personal information management data, including address book, diary).

[0467] a computer implemented dynamic language model sub-system adapted to build a dynamic language model using one or more of: caller dependence; call-pair dependence; callee dependence.

[0468] The caller is anyone depositing a voice message, irrespective of whether they intend to place a voice call; and the callee is anyone who reads the converted message, irrespective of whether they were meant to receive a voice call.

[0469] a computer implemented personal profile sub-system adapted to build a personal profile of a caller to improve conversion accuracy.

[0470] The personal profile includes words, phrases, grammar, or intonation of the caller.

[0471] Core Concept D

[0472] 3 Part Message Taxonomy

[0473] A mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen; the system comprising (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising:

[0474] a computer implemented boundary selection sub-system adapted to process a message by looking for the boundaries between sections of the message which carry different types of content or carry different types of message.

[0475] Other features:

[0476] the computer implemented boundary selection sub-system analyses for one or more of the following component parts: a greeting part; a body part; a goodbye part.

[0477] different conversion strategies are applied to each part, the applied strategy being optimal for converting that part.

[0478] different parts of the message have different quality requirements and a quality assessment sub-system applies different standards to those different parts.

[0479] A speech quality estimator detects boundaries between sections of the message which carry different types of content or carry different types of message.

[0480] Boundaries are detected or inferred at regions in the message where the speech density alters.

[0481] Boundaries are detected or inferred at a pause in the message.

[0482] Boundaries are inferred as arising at a predefined proportion of the message.

[0483] A greeting boundary is inferred at approximately 15% of the entire message length.

[0484] Core Concept E

[0485] Pre-Processing Front-End

[0486] A mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen; the system comprising (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising:

[0487] a computer implemented pre-processing front-end sub-system that determines an appropriate conversion strategy used to convert the voice messages.

[0488] Other features:

[0489] The pre-processing front end optimises the quality of the audio for conversion by performing one or more of the following functions: removing noise, cleaning up known defects, normalising volume/signal energy, removing silent/empty sections and classifies the message type for optimally routing the message for conversion, or not.

[0490] The pre-processing front-end determines the language being used by the caller, based on one or more of the following: knowledge about registration, location and call history of caller and/or receiver.

[0491] The pre-processing front-end selects a particular ASR engine to convert a message or part of a message.

[0492] different conversion resources, such as ASR engines, are used for different parts of the same message.

[0493] different conversion resources, such as ASR engines, are used for different messages.

[0494] the human operators are treated as ASR engines.

[0495] the pre-processing front-end uses or is connected with a recogniser confidence sub-system to determine automatically the confidence level associated with a conversion of a specific message, or part of a message, and a particular conversion resource, such as an ASR engine, is then deployed depending on that confidence level.

[0496] the conversion strategy involves the selection a conversion strategy from a set of conversion strategies that include the following: (i) messages for which an ASR conversion confidence is sufficiently high are checked automatically by a quality assessment sub-system for conformance to quality standards; (ii) messages for which the ASR conversion confidence is not sufficiently high are routed to a human operator for checking and, if necessary, correction; (iii) messages for which the ASR conversion confidence is very low are flagged as unconvertible and the user is informed of the receipt of an unconvertible message.

[0497] Core Concept F

[0498] Queue Manager

[0499] A mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen; the system comprising (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising:

[0500] a computer implemented queue manager sub-system that intelligently manages loading and calls in resources as required to ensure that converted message delivery times meet a pre-defined standard.

[0501] Other features:

[0502] The queue manager sub-system determines what should happen to a voice message at each stage of processing through the system.

[0503] If at any stage of the automated conversion, confidence intervals or other measures suggest any part of a message are not good enough, then the queue manager directs this to the correct human operator for assistance.

[0504] The queue manager sub-system makes decisions by calculating trade-offs between conversion time and quality.

[0505] The queue manager sub-system uses state-machines that, for any given language queue, can decide how best to process the messages through the system.

[0506] Core Concept G

[0507] Lattice

[0508] A mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen; the system comprising (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising

[0509] a computer implemented lattice sub-system that generates a lattice of possible word or phrase sequences and enables a human operator to guide a conversion sub-system by being shown one or more candidate converted words or phrases from the lattice and enabling the operator to either select that candidate word or phrase or, by entering one or more characters for a different converted word or phrase, to trigger the conversion sub-system to propose an alternative word or phrase.

[0510] Other features:

[0511] The conversion sub-system receives inputs from a sub-system that handles call-pair history information.

[0512] The conversion sub-system receives inputs from conversion resources.

[0513] The conversion sub-system receives inputs from a context sub-system that has knowledge of the context of a message.

[0514] The conversion sub-system learns, from the human operator inputs, likely words corresponding to a sound pattern.

[0515] The human operator is required to select only a single key to accept a word or phrase.

[0516] The conversion sub-system automatically provides capitalisation and punctuation.

[0517] The conversion sub-system can propose candidate numbers, real nouns, web addresses, e-mail addresses, physical addresses, location information, or other coordinates.

[0518] The conversion sub-system automatically differentiates between parts of the message that are likely to be important and those that are likely to be unimportant.

[0519] Unimportant parts of the message are confirmed by the operator as belonging to a class proposed by the conversion sub-system and are then converted solely by a machine ASR engine.

[0520] The human operator can speak the correct word to the conversion system, which then automatically transcribes it.

[0521] Core Concept H

[0522] On-Line Corpus

[0523] A mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen; the system comprising (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising:

[0524] a computer implemented search sub-system that analyses a converted message against an on-line corpus of knowledge.

[0525] Other features:

[0526] The on-line corpus of knowledge is the internet, as accessed by a search engine.

[0527] The on-line corpus of knowledge is a search engine database, such as Google.

[0528] The analysis of the converted message enables the accuracy of the conversion to be assessed by a human operator an/or a recogniser confidence sub-system.

[0529] The analysis of the converted message enables ambiguities in the message to be resolved by a human operator and/or an ASR engine.

[0530] Core Concept I

[0531] Detectors

[0532] A mass-scale, user-independent, device-independent, voice messaging system that converts unstructured voice messages into text for display on a screen; the system comprising (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising:

[0533] a computer implemented detector sub-system that is adapted to detect slam-downs.

[0534] Other features:

[0535] The slam down detector is implemented as part of a pre-processing front-end.

[0536] Other Detectors that can Also be Used:

[0537] A computer implemented detector sub-system that is tuned to detect different spoken languages, such as English, Spanish, French etc.

[0538] The language detector can detect changes in language part way through a message.

[0539] The language detector can use inputs from a sub-system that has call-pair history information that records how changes in language occurred in earlier messages.

[0540] A computer implemented detector sub-system that is adapted to estimate speech quality

[0541] The speech quality estimator finds drop-outs, estimate noise levels and calculate an overall measure of voice quality and uses an adaptive threshold to reject lowest quality messages.

[0542] A computer implemented detector sub-system that is adapted to detect slam-downs.

[0543] The slam down detector is implemented as part of a pre-processing front-end.

[0544] A computer implemented detector sub-system that is adapted to detect inadvertent calls.

[0545] The inadvertent call detector is implemented as part of a pre-processing front-end.

[0546] A computer implemented detector sub-system that is adapted to detect and convert pre-recorded messages.

[0547] A computer implemented detector sub-system that is adapted to detect and convert spoken numbers.

[0548] A computer implemented detector sub-system that is adapted to detect and convert spoken addresses.

[0549] A computer implemented detector sub-system that is adapted to detect and convert real nouns, numbers, web addresses, e-mail addresses, physical addresses, location information, other coordinates.

[0550] Message Types

[0551] The message is a voicemail intended for a mobile telephone and the voice message is converted to text and sent to that mobile telephone.

[0552] The message is a voice message intended for an instant messaging service and the voice message is converted to text and sent to an instant messaging service for a display on a screen.

[0553] The message is a voice message intended for a web blog and the voice message is converted to text and sent to a server for display as part of the web blog.

[0554] The message is a voice message intended to be converted to text format and sent as a text message.

[0555] The message is a voice message intended to be converted to text format and sent as an email message.

[0556] The message is a voice message intended to be converted to text format and sent as a note or memo, by email or text, to an originator of the message.

[0557] Other Elements of the Value Chain

[0558] A mobile telephone network that is connected to the system of any preceding Claim.

[0559] A mobile telephone when displaying a message converted by the system of any preceding Claim.

[0560] A computer display screen when displaying a message converted by the system of any preceding Claim.

[0561] A method of providing voice messaging, comprising the step of a user sending a voice message to a messaging system as claimed in any preceding Claim.

1. A mass-scale, user-independent, device-independent, voice messaging system that converts an unstructured voice message into text for display on a screen; the system comprising (i) computer implemented sub-systems and also (ii) a network connection to human operators providing transcription and quality control; the system being adapted to optimise the effectiveness of the human operators by further comprising:

a computer implemented pre-processing front-end sub-system that determines an appropriate conversion strategy used to convert the voice message.

2. The system of claim 1 in which the pre-processing front end optimises the quality of the audio for conversion by removing noise.

3. The system of claim 1 in which the pre-processing front end optimises the quality of the audio for conversion by cleaning up known defects.

4. The system of claim 1 in which the pre-processing front end optimises the quality of the audio for conversion by normalising volume/signal energy.

5. The system of claim 1 in which the pre-processing front end optimises the quality of the audio for conversion by removing silent/empty sections and classifies the message type for optimally routing the message for conversion, or not.

6. The system of claim 1 in which the pre-processing front-end determines the language being used by the caller, based on one or more of the following: knowledge about registration, location and call history of caller and/or receiver.

7. The system of claim 1 in which the pre-processing front-end selects a particular ASR engine to convert a message or part of a message.

8. The system of claim 1 in which different conversion resources, such as ASR engines, are used for different parts of the same message.

9. The system of claim 1 in which different conversion resources, such as ASR engines, are used for different messages.

10. The system of claim 1 in which the human operators are treated as ASR engines.

11. The system of claim 1 in which the pre-processing front-end uses or is connected with a recogniser confidence sub-system to determine automatically the confidence level associated with a conversion of a specific message, or part of a message, and a particular conversion resource, such as an ASR engine, is then deployed depending on that confidence level.

12. The system of claim 1 in which the conversion strategy involves the selection a conversion strategy from a set of conversion strategies that include the following: (i) messages for which an ASR conversion confidence is sufficiently high are checked automatically by a quality assessment sub-system for conformance to quality standards; (ii) messages for which the ASR conversion confidence is not sufficiently high are routed to a human operator for checking and, if necessary, correction; (iii) messages for which the ASR conversion confidence is very low are flagged as unconvertible and the user is informed of the receipt of an unconvertible message.

13. The system of claim 1 in which the message is a voicemail intended for a mobile telephone and the voice message is converted to text and sent to that mobile telephone.

14. The system of claim 1 in which the message is a voice message intended for an instant messaging service and the voice message is converted to text and sent to an instant messaging service for a display on a screen.

15. The system of claim 1 in which the message is a voice message intended for a web service and the voice message is converted to text and sent to a server for display as part of the web service.

16. The system of claim 1 in which the message is a voice message intended to be converted to text format and sent as a text message.

17. The system of claim 1 in which the message is a voice message intended to be converted to text format and sent as an email message.

18. The system of claim 1 in which the message is a voice message intended to be converted to text format and sent as a note or memo, by email or text, to an originator of the message.

19. A mobile telephone network that is connected to the system of claim 1.

20. A mobile telephone when displaying a message converted by the system of claim 1.

21. A computer display screen when displaying a message converted by the system of claim 1.

22. A method of providing voice messaging, comprising the step of a user sending a voice message to a messaging system as claimed in claim 1.

* * * * *