A Natural Language Interface to a Theater Information Database

Margus Treumuth[1], Tanel Alumäe[2], and Einar Meister[2]

[1] Institute of Computer Science
University of Tartu, Tartu, Estonia
treumuth@ut.ee
[2] Institute of Cybernetics
Tallinn University of Technology
Tallinn, Estonia
{tanel.alumae, einar}@phon.ioc.ee

Abstract

The development of a natural language dialogue system as an interface to a theater information database is a joint research project of the University of Tartu (Estonia) and the Tallinn University of Technology (Estonia). The underlying database contains information about theater performances in a certain theater or city. The dialogue system can be used to ask information about performances using either spoken or typewritten natural language in Estonian. The dialogue management module was developed at the University of Tartu while the modules for speech recognition and speech synthesis were added by the Tallinn University of Technology. This article discusses the development of the dialogue module, the speech recognition module, and the speech synthesis module.

Sistem za dialog v naravnem jeziku kot vmesnik do gledališke podatkovne baze

Razvoj sistema za dialog v naravnem jeziku kot vmesnik do gledališke podatkovne baze je skupni raziskovalni projekt Univerze v Tartuju (Estonija) in Tehniške univerze v Talinu (Estonija). Podatkovna baza vsebuje informacije o predstavah v določenem gledališču ali mestu. Sistem za dialog je mogoče uporabiti za pridobivanje informacij o predstavah bodisi v govorjeni ali pisni estonščini. Modul za vodenje dialoga je bil razvit na Univerzi v Tartuju, medtem ko so module za razpoznavanje in sintezo govora dodali na Tehniški univerzi v Talinu. Prispevek obravnava razvoj modulov za dialog, razpoznavanje govora in sintezo govora.

1. Introduction

The dialogue system developed in this project operates in a constrained linguistic domain – theater information. The underlying database contains information about theater performances in a certain theater or city. The dialogue system can be used to ask information about performances. Currently, the system does not contain price or booking information, also the names of actors and authors are not included at this time. The system can deal with either typewritten or spoken language. The typewritten interface is accessible at http://www.dialoogid.ee/. The language used by the dialogue system is Estonian.

The research groups involved are with the Institute of Cybernetics at the Laboratory of Phonetics and Speech Technology of the Tallinn University of Technology, and with the Institute of Computer Science at the Research Group of Computational Linguistics at Tartu University.

The paper is organized as follows. Section 2 describes the system components. Section 3 is concerned with experiments. Section 4 describes the software environment of the implementation. Finally, our conclusions are stated in section 5.

2. System Components

The dialogue system consists of modules for speech recognition, dialogue management, morphological analysis, query generation and speech synthesis (see Figure 1). The date recognition module is a submodule of query generation The theater information is stored in a relational database system. The speech recognition, dialogue management and speech synthesis modules are all run as autonomous services

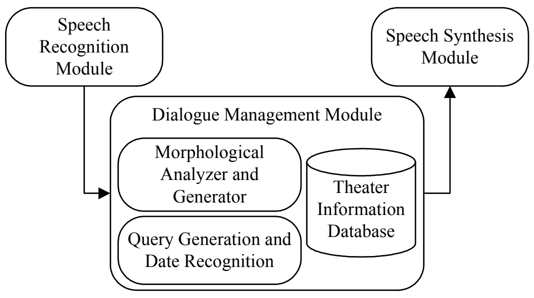
Speech Recognition Module
Speech Synthesis Module
Dialogue Management Module
Morphological Analyzer and Generator
Theater Information Database
Query Generation and Date Recognition

Figure 1: Dialogue system architecture

2.1. Speech Recognition Module

The speech recognition module segments the input stream into utterances and produces a recognition hypothesis for each segment. It also triggers barge-in, if it detects speech that continues for a configurable amount of time. Barge-in sends a signal to the speech synthesis module to stop any speech output.

2.1.1. Acoustic Modeling

The acoustic models for recognition experiments were trained on the Estonian SpeechDat-like phonetic database (Meister, et al., 2003), collected from volunteer speakers over the telephone network. The total number of different speakers in the database is 1332. The number of acceptable utterances is 177 793. This represents about 241.1 hours of audio data.

The speech data was recorded at an 8 kHz sampling rate and coded using 8-bit mono A-law. The recording sessions consisted of a fixed set of utterance types, such as isolated and connected digits, natural numbers, monetary amounts, spelled words, time phrases, date phrases, yes/no answers, person and company names, application words and phrases, phonetically rich words, and sentences.

The open source SphinxTrain toolkit was used for training the acoustic models. Models were created for 25 phonemes, the five filler/noise types and silence. For acoustic features, MFCC coefficients were used. The coefficients were calculated from a frequency band ranging from 130 Hz to 3400 Hz, using a pre-emphasis coefficient of 0.9. The window size was 0.0256 seconds and the frame rate was 100 frames/second. A 512-point FFT was used to calculate 31 filter banks, out of which 13 cepstral coefficients were generated. All units are modeled by continuous left-to-right HMMs with three emitting states and no skip transitions. The output vectors are 39-dimensional and are composed of 13 cepstral coefficients, delta and double delta coefficients. The final tied-state triphone models have 8000 shared states in total. Each state is modeled by eight Gaussian mixture components.

The pronunciation dictionary is automatically created from word orthography using a set of context sensitive rewrite rules. Since many performance names contain foreign names, there is an additional manually compiled pronunciation dictionary of non-native words that is merged to the rule-driven dictionary.

2.1.2. Language Modeling

For speech recognition language modeling, a class-based trigram model is used. The training data for the language model is a set of sample questions to the system, composed by system developers and testers, and collected during live system testing. Some of the word classes used in the language model are: city names, theater names, performance names (synchronizable with the dialogue manager database), day-of-month names, month names, and weekdays. As Estonian is an inflective language, most such words can occur in many inflections. Thus, there are separate classes for each common inflection, e.g., [weekday, nom. sg.], [weekday, gen. sg.], [weekday, ad. sg.] as in *esmaspäev*, 'Monday' nom. sg.; *esmaspäeva*, 'Monday' gen. sg.; and *esmaspäeval*, 'Monday' ad. sg.

During the training process, all words in the training sentences that belong to any class are replaced with the corresponding class tag. The resulting pseudo-sentences were used to train the trigram language model. All intra-class probabilities were distributed evenly.

One problem with the language model is the common use of shortened performance names in user queries. For example, instead of saying the full name "Pianola or The Mechanical Piano" (a popular performance), users tend to refer to just "Pianola", often using the inflected word form in the sentence (e.g., "*Millal mängitakse Pianolat?*", "When is Pianola (gen. sg.) being played?"). Such shorthand names are difficult to predict automatically from the performance database. To cope with this, we manually composed a list of such short names and put them to a separate class (actually two classes - one for the nominative and one for the often occurring genitive case). However, those classes must be manually checked from time to time for new entry candidates which creates some additional administrative burden.

2.2. Dialogue Management Module

The dialogue management module integrates an Estonian morphological analyzer/generator (Kaalep, 1997). The Estonian language is an agglutinative language that is rich in morphology. Therefore, the parsing technique involves automatic generation of lemmas or base forms using morphological analyzer. This way the system can handle minor deviations of the input. As a weakness – the morphological base form generation could also trigger ambiguity leading to unexpected results. Some deviations still cause problems and in the near future we will use the spell checking functions of the morphological analyzer to handle typing errors. We also plan to use the Levenshtein algorithm to calculate the distance between strings. This way we can guess which word (from a dictionary or database) is intended when an unknown word is encountered.

The morphological generator is used to produce required forms of words from base forms e.g. from *jaanuar* (January nom. sg.) to *jaanuaril* (on January ad. sg.).

The system also uses examples of linguistic phenomena from a dialogue corpus (Gerassimenko, et al., 2004), yet it is not a stochastic approach as probabilistic techniques are not used at the moment. We examined the corpus to see how the users phrase their questions and how they express dates and time.

2.2.1. Knowledge Base

The knowledge base of the dialogue management module consists of a primary database and a secondary database. The primary database contains the theater information (city, theater, performance, date) and the secondary database contains some simple linguistic facts (domain specific words by attribute-value pairs), e.g.:

Keyword (the knowledge base is in Estonian): *pilet* (ticket nom. sg.)

Values for keyword *pilet* (the number of values per keyword is not limited):

Piletite hinnad puuduvad. (Sorry, we have no ticket prices).

Piletite kohta kahjuks info puudub. (Sorry, we have no information about tickets).

Tean ainult etenduste algusaegasid. (We only know the dates of performances).

The primary database – the database of theaters and performances – is gathered manually from several online databases and is also updatable using a web interface. We also plan to arrange the database to run automatic updates daily.

The secondary database is also directly visible and adjustable in a knowledge base settings file. Various forms of greeting expressions and some domain specific phrases can be modified by a system administrator.

2.2.2. Query Generation Module

The query generation component converts user input to SQL (structured query language) queries - commands to be passed to the underlying database.

The primary parser detects proper names that occur in the primary relational database (names of performances and theaters), date and time phrases. Once a certain keyword has been recognized, the system may retrieve the answer from the database having the power of SQL available for quick definition and manipulation of data.

Let us consider a dialogue taking place between a human and a computer.

<Human>: I would like to see the musical *Cats* on *March 17*.
<Computer>: Cats is not playing on March 17. It is playing on March 19.

In this example, the highlighted words (Cats and March 17) are the actual keywords which convey the most important information and are recognized by the system. All the other words are semantically irrelevant and can be ignored. The inflections that occur in Estonian language, as in *märtsini* (March ter. sg.), *märtsil* (March ad. sg.), are handled by the morphological analyzer that generates the base form: *märts* (March nom. sg.).

The previous example might seem simple, yet the recognition of dates is not a simple task. There are many ways users can express dates and time (e.g. next Friday, on Christmas day, two weeks from today). Therefore, we have created a separate module in our system for date recognition.

The reaction to a user utterance depends on the state of the dialogue (dialogue context). That is, the choice of answer is based on previously acquired knowledge. Users can continue to ask queries about the previous topic. The system can remember facts the user has asked before. For example, if the user has mentioned a theater by name, all further references to some certain dates are handled in the context of the theater mentioned previously, e.g.,

<Human>: What is playing at Theater Royal?
<Computer>: The Producers is playing today.
<Human>: What about tomorrow?
<Computer>: There are no plays at the Theater Royal tomorrow.

The secondary parser is used if the primary parser gives no results. It can recognize only predefined words and/or phrases described in the secondary database and can only respond using a number of predefined sentence patterns also described in the secondary database.

Randomization is used in choosing the answer from the secondary database to provide the effect of non-linear transformations between inputs and outputs. Users tend to like slight unexpectedness and surprises (e.g., various expressions of greetings). Users will get bored if the system is too predictable and determine the limits of the system too quickly.

It is essential to keep users actively engaged in trying to get the answers they are searching for. This will provide the developers with valuable chat logs as the system stores all conversations. These chat logs are later used as training data to manually improve the performance of the linguistic model that relies on collection of predefined keywords and expressions to represent semantic notions.

2.3. Speech Synthesis Module

Search results can be presented to the user in two modalities: in text form and/or via speech output. In the latter case the written answer will serve as the input text for an Estonian text-to-speech synthesizer (Mihkla, et al., 1999).

Speech synthesis starts with the linguistic analysis of the input text, where the orthographic text is converted into phonemic representation. The linguistic module identifies numbers, abbreviations and acronyms in the input sentence and transforms them into full orthographic text. Next, the orthographic text is converted into an adequate phonemic representation. A prosody model calculates the phoneme durations and the contour of fundamental frequency according to the communicative type of sentence. Phonemic and prosodic information serve as input for the acoustic unit generation, which is based on the concatenative MBROLA model (Dutoit, et al., 1993). The MBROLA-engine utilizes diphones as the elementary concatenative units; the Estonian diphone database includes about 1700 diphones.

Using synthetic speech as the output of a dialogue system presents high demands on the prosodic (especially intonation) modeling – the spoken answer must be adequate with the dialogue structure and prosodically suit the on-going discourse. The current version of the text-to-speech synthesizer relies only on the linguistic information of the input text and is not able to model the prosodic structure of dialogue speech. Therefore, the speech output is produced almost identically for different types of answers. Significant improvement in prosody modeling could be achieved by including information about dialogue structure.

3. Results of Experiments

The system was tested with 150 conversations, some typewritten and some spoken. There were differences in typewritten and spoken conversations, yet the distinction between those is not important at this stage of development.

The system failed 25% of the time when attempting to answer a question. Yet, the subjects failed to communicate successfully with the system only 5% of the time. This shows that the system can make mistakes while users are still able to get their answers by rephrasing their questions.

The main problems are:

- Some errors occur in pattern matching when minor deviations in the input take place. These occur mainly while matching the names of performances or names of theatres. As stated above, in the near future we will apply the Levenshtein algorithm to calculate the distance between strings. This way we can guess which word (from a dictionary or database) is meant when an unknown word is encountered. We also plan to use the spell checking functions of the morphological analyzer to handle typing errors.
- The users quickly discover the limits of the system's knowledge – there is no information about ticket prices, no booking. So the user is unable to retrieve this information from the system. We plan to expand the knowledge base and include ticket prices and booking information in the near future.

4. Implementation

The dialogue module is a web enabled system. Therefore, it is easy for the developer to add, modify and deploy new functionality. The web enabled system also provides an easy way to collect chat logs that can be used as training data to manually improve the performance of the linguistic model.

The dialogue module was developed in PHP using MySQL as the database server and Apache HTTP Server as the web server.

The speech recognition and speech synthesis modules are standalone modules developed by collaborating researchers at the Tallinn University of Technology.

5. Conclusions

The dialogue system developed in this project demonstrates that we are capable of designing systems which can understand a small subset of natural language in a constrained linguistic domain. The conducted experiments show that our methods are satisfactory yet need some further improvements.

In the future, we hope to release a dialog based speech-understanding system that could be used over a telephone.

6. Acknowledgements

The project has support from the Estonian Ministry of Culture, the Estonian Information Technology Foundation, the Estonian Science Foundation (grant no 5685), and Elion Enterprises Ltd.

7. References

Gerassimenko, O., Hennoste, T., Koit, M., Rääbis, A., Strandson, K., Valdisoo, M., Vutt, E. (2004). Annotated dialogue corpus as a language resource: an experience of building the Estonian dialogue corpus. In: The first Baltic conference “Human language technologies. The Baltic perspective”. Commission of the official language at the chancellery of the president of Latvia, Riga 150–155

Kaalep, H.-J. (1997). An Estonian Morphological Analyser and the Impact of a Corpus on Its Development. Computers and the Humanities 31: 115-133

Meister, E., Lasn, J., Meister, L. (2003). SpeechDat-like Estonian database. - In: Text, Speech and Dialogue : 6th International Conference, TSD 2003, Czech Republic, September 8-12, 2003 / Eds. Matoušek [et al.]. Berlin [etc.] : Springer, Lecture Notes in Artificial Intelligence, Vol. 2807. 412-417

Mihkla M., Eek A., Meister E. (1999). Text-to-Speech Synthesis of Estonian. – Proceedings of the 6th European Conference on Speech Communication and Technology, Budapest, Vol. 5 2095-2098