

Robust Content Extraction for Translation and Dialog Processing

Norbert Reithinger and Ralf Engel

DFKI GmbH, Saarbrücken, Germany

Abstract. The design rationale guiding the development of the reductionist dialog act based translation module in Verbmobil was robustness. Even in case the speech recognition or the prosodic processing does not perform perfectly, this module extracts and translates the main intentions and facts related to the domain. In a three step approach, first the dialog act describing the intention is computed using a statistical approach. The second step is the construction of the propositional content with robust hierarchical finite state transducers. For the definition of the transducers, knowledge sources available in Verbmobil are exploited. The resulting representation of these two steps is used in a template based finite state generator to realize the target language expressions. The internal representation is also communicated to the dialog module where it plays an important part in maintaining the dialog state.

1 Overview

The dialog act based approach to translation we present in this chapter has its origin in the first phase of Verbmobil. Soon after the beginning of the project it became clear that Verbmobil needed a translation track that delivers a translation robustly, restricted to the main intention and basic facts of the domain. Even under adverse conditions in the input, e.g., weak word and boundary recognition, a module was needed that provides this type of processing behavior.

The first module in the first phase of Verbmobil that was based on this idea was joint work by Siemens AG, University of Erlangen-Nürnberg, and DFKI GmbH, Saarbrücken (Block, 1997). In the second phase of Verbmobil, we redesigned and reimplemented the module, called `syndialog` in the running system, in the spirit of the first approach, however using different approaches as described below.

The dialog act based translation consists of three steps

- extract the core intention, described by the dialog act
- extract the content representation and use it as a restricted interlingua
- generate the target language utterance from this representation

A turn is not translated as one unit, but it is split into smaller segments using the prosody module (see Chapter ??, page ??). Each of these segments is translated separately.

In the remainder of this chapter we will present all three steps. The system is implemented to translate between all three languages used in Verbmobil. The content representation is also used in the dialog module of Verbmobil to create a representation of the ongoing dialog, which is exploited for the dialog summary and dialog minutes generation.

2 Dialog Act Recognition

2.1 The Used Dialog Acts

Intentions in Verbmobil are described by means of dialog acts (see Chapter ??, page ?? for details). Our dialog act scheme defines 35 dialog acts, structured in a hierarchy (Alexandersson et al., 1998). We annotated 21 CD-ROMs of the Verbmobil corpus with these acts to be used for the computation of dialog acts.

For processing purposes we use only 19 acts¹. Utterances annotated with the other 16 acts are mapped on the 19 used acts exploiting the hierarchy. The reasons for this approach are twofold: firstly, 10 of the dialog acts combined cover less than 1% of the annotated utterances. They contain acts like `DEVIATE_SCENARIO` that neither carry propositional content central to negotiation dialogs nor do they control the dialog like the act `DEFER` which occurs also not very frequently but has an important function in the progress of a dialog. The other 6 acts are frequently confused with closely related acts, e.g., `FEEDBACK_POSITIVE` with `ACCEPT` where the latter contains a propositional content, whereas the first one does not. We developed tools to create and analyze confusion matrices amongst others to detect such cases, and to finally arrive at the dialog act mapping to the reduced set of 19 acts.

2.2 The Recognition Method

In the past years, mainly three methods for classification of dialog acts were proposed and applied to larger corpora: statistical classifiers, using language models (see, e.g., Mast et al., 1995, Reithinger and Klesen, 1997, Tanaka and Yokoo, 1999, Choi et al., 1999), neural networks (Kipp, 1998), and transformation based learning (Samuel et al., 1998).

After evaluating all three approaches, we use a statistical classifier (Reithinger and Klesen, 1997). It decides which dialog act D describes the illocution of a certain string of words W with statistical models for each D , under the condition of a given W . It selects the model that is most probable, i.e.

$$D = \operatorname{argmax}_{D'} P(D'|W)$$

which can as usual be reformulated using Bayes' rule to

¹ The Japanese dialogs were annotated using the second revision of our dialog act annotation scheme. It did not include the dialog acts `CLOSE`, `COMMIT`, `DEFER`, `INFORM_FEATURE`, and `REQUEST_COMMIT`.

$$D = \operatorname{argmax}_{D'} P(W|D') P(D')$$

We can exploit the knowledge about the previous dialog history H by using $P(D'|H)$ from a prediction component (Reithinger et al., 1996) as a statistical dialog model. Experiments show that correct classifications are up to 3% better if we include this additional knowledge. In the experiments presented below, recognition rates increase by 1–2% when using the dialog history. Therefore, we use

$$D = \operatorname{argmax}_{D'} P(W|D') P(D'|H)$$

The a-priori probabilities $P(W|D')$ and $P(D'|H)$ can be approximated from the annotated Verbmobil corpus. All utterances for one dialog act are collected and the relative word and transition frequencies are used for the approximation of $P(W|D')$. The dialog act transition probability $P(D'|H)$ is also approximated by the relative transition frequencies of the acts in the dialogs.

Various methods are proposed to approximate the distributions from the conditional frequencies. We implemented a flexible classifier workbench in LISP and augmented it with evaluation and visualization tools to analyse the results. Using this workbench, we tested algorithms (Klesen, 1997) like linear and rational interpolation, backing-off or using the multi variant Poisson distribution. The best results were provided when using a linear n -gram interpolation using equally distributed interpolation weights. The optimization of the weights using the EM-algorithm on an evaluation set usually reduces perplexity for this set but yields also an over-adaptation to it.

2.3 Recognition Rates in Leave One Out Experiments

Most approaches in dialog act recognition (Mast et al., 1995, Samuel et al., 1998, Choi et al., 1999) partition their corpus in fixed sets of training and test dialogs. However, the selection of the test/training partition might influence the results of the evaluation. Therefore, we made extensive leave one out experiments. In these experiments, we trained language models for each of the 1505 dialogs in the annotated corpus we use, where only the test dialog was held back from the training data.

Table 1 shows detailed recall and precision values for the used dialog acts. We show the overall results of a whole leave one out experiment testing all 1505 dialogs. The acts expressed by highly standardized phrases like GREET are recognized with high accuracy. Problematic acts are, e.g., CLOSE, where only few training samples are available. Pairs of acts that are confused frequently with each other, like GIVE_REASON and REJECT, can be identified using confusion matrices.

To see how the length of our dialogs measured in dialog acts is related to recall, we set up another leave one out experiment, using bigram models. In summary, it shows that short dialogs have usually a higher recall. For German, the maximum of over 80% is reached for dialogs with about 20 acts. Recall drops below 70% when

Table 1. Results with bigram word interpolation for all languages

<i>dialog act</i>	<i>recall (%)</i>	<i>precision (%)</i>
INTRODUCE	97.45	95.37
THANK	96.05	94.13
BYE	94.53	91.31
GREET	94.26	91.77
ACCEPT	85.34	78.97
INIT	74.23	68.42
SUGGEST	72.20	66.23
POLITENESS_FORMULA	71.66	78.94
COMMIT	69.04	61.35
REJECT	68.75	64.24
REQUEST_SUGGEST	64.29	64.79
INFORM_FEATURE	63.35	45.62
REQUEST_COMMENT	59.32	66.76
CLOSE	52.03	34.56
GIVE_REASON	42.85	51.24
INFORM	42.74	55.68
DEFER	41.01	41.20
REQUEST_COMMIT	39.78	63.79
REQUEST	30.71	46.01
<i>overall</i>	68.61	66.34

the dialog length is greater than 80 acts. For English dialogs up to the mean length of 60 acts have a recall above 75%, which drops for longer dialogs. Japanese shows more variation. Recall drops below 75% with dialogs consisting of 31 dialog acts and more.

We see various reasons for this results: short dialogs in our corpus are usually very straightforwardly centered on the tasks and domains, and have less deviations. The majority of the training data is relatively short in all three languages. Overall recognition peaks for those dialogs, where most comparable training material is available — a fact that is to be expected. These short dialogs also do not include longer stretches of pure information exchanges. Those utterances labeled with INFORM are recognized poorly, even if they are rather frequent in the corpus and therefore in the training sets.

2.4 Comparison

A comparison of different approaches for dialog act classification is currently almost impossible. The only approaches that were trained and tested on exactly the same data and tag set of 18 dialog acts are (Reithinger and Klesen, 1997) and (Samuel et al., 1998). The first report an overall recall value of 74.7%, the latter of 73.1% for an English corpus. The neural network approach in (Kipp, 1998) performed slightly worse than a statistical n -gram model approach on the same German corpus, also

using 18 dialog acts. Other statistical approaches like (Choi et al., 1999) who report a 81.6% recall and use a Korean corpus with 19 dialog acts, differ in language, corpus size, or tag set. All of these evaluations use a fixed set of dialogs for training and testing. To our knowledge, there exists no other leave on out evaluation for dialog act recognition.

3 Content Extraction

3.1 Representation of Propositional Content

After determining the dialog act the second part is the extraction of the propositional content. The content representation consists of nested objects, a modified subset of the propositional representation formalism used in the context evaluation module (see Chapter ??, page ??) and similar to approaches used in C-Star (Corazza, 1999). The design of the objects was developed by examining the Verbmobil corpus (see Chapter ??, page ??). Most of the relevant information for the topics scheduling, traveling, accommodation and entertainment is covered. 49 different object types and 95 different attributes exist.

An example of the propositional structure for the utterance

“We could leave Hamburg on the first and I could book two tickets.”

including dialog act and topic is

```
[SUGGEST, traveling, has_book_action:[book_action,
  has_move:[move,
    has_source_location:[city, has_name='hamburg'],
    has_departure_time:[date, time=[day:1]],
    has_ticket:[ticket, has_number=2]]]
```

3.2 The FST Approach to Robust Processing

To extract the content of an utterance, finite state transducers (FST) (Appelt et al., 1993) are used. We chose FSTs because they are fast, work robustly, are better manageable than plain code, and easy to implement. The FSTs are ordered hierarchically, i.e., the output of one FST is the input for the next FST. Since the propositional representation consists of nested objects, the FST engine actively supports the creation and modification of objects. The input and output of a FST can be a mixture of natural language words and created objects.

A couple of tools support the development, e.g., a drawing tool for FSTs implemented in Tcl/Tk (see Figure 1), a syntax checker for the representation, trace tools to examine the effects of changes within the FSTs, and a tool for merging FSTs, supporting distributed development.

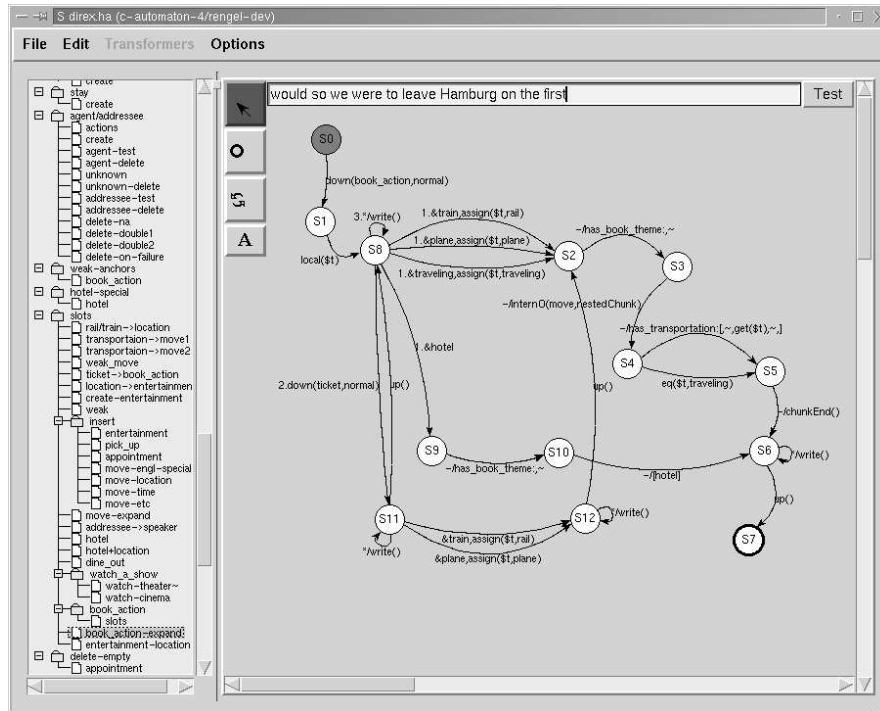


Figure 1. The graphical FST editor

3.3 Building the Content Representation

The extraction task can roughly be divided into three subtasks. These are described in the following including some of the occurring problems.

The first step is to extract temporal expressions. A set of 136 FSTs is responsible for transforming spoken temporal expressions into the formal equivalents (see Chapter ??), including complex constructions. Temporal expressions are sometimes hard to detect, e.g., *should we meet at five* or *the first sounds good*, where no explicit *five o'clock* or *first of August* is present. The (not perfect) solution chosen is to mark these time objects as uncertain and decide later when the objects are merged, if in this context a time expression is probable or not, or even if a number belongs to an extracted non-temporal object (like in *I book two tickets*).

Creating simple objects using keyword-spotting is the next step in the process chain. Task relevant words are transformed into simple objects, e.g., locations like *Hamburg* (`[city, has_name = 'hamburg']`) or verbs like *travel* (`[move]`). We are using simple but robust keyword-spotting instead of more complex methods to avoid failures on higher word recognition error rates. Of course the content words

have to be correctly recognized. A drawback of this approach is that complex sentences, e.g., if-then-sentences, are often analysed incorrectly.

The lexical database LexDB and the semantic database SemDB of Verbmobil (see Chapter ??, page ??) support word spotting and object creation and are used for the automatic generation of basis FSTs. The LexDB allows to mention only the stem in the FSTs, the derived forms are added automatically, a feature especially useful for German. In the SemDB semantic information is encoded, e.g., the words that denote locations, or verbs which are related to traveling.

One problem is that many words are ambiguous in their meaning, e.g., *leave* needs other objects associated with traveling to trigger a `move` object. Sometimes cautious decisions have to be made and lead to a loss of information. It is also not always clear what type of object a word triggers, e.g., *bar* can trigger one of `part_of_hotel`, `location`, or `entertainment_location`. This problem is solved by choosing one possibility and adding hints to the according object. Later the object type is changed if required by the context. Hints are also used to save information which is necessary for determining the role during embedding (e.g., *to Munich* becomes `has_dest_location: [city, has_name= 'muenchen']`).

Combining the extracted simple objects into complex ones is the last step, e.g., embedding `[city, has_name= 'hamburg']` in `[move]`. This can be iterated, e.g., the `move` object is embedded in a `book_action` object. Special care is required for enumerations like several hotels and prices occurring in one utterance where each hotel has one price. Another problem arises when different topics are mentioned in one utterance and it is unclear for some of the objects where they should be embedded.

In all three subtasks the solutions finally chosen for problems are often a compromise between increasing precision and increasing recall. Most of the time we chose a conservative approach to get higher precision. We defined 334 multi-language FSTs for the analysis of German, English and Japanese. The FSTs were empirically derived from about 30 000 utterances of the Verbmobil corpus.

4 Generation in the Target Languages

After the content structure is built up from an utterance in the source language, an utterance in the desired target language has to be generated. The core of our approach is a set of sentence templates combined with FSTs responsible for the language generation of the content of the embedded objects. 149 sentence templates for each language and 30 multi-lingual generation FSTs for the different object types exist. A confidence value, required for the selection module, is calculated considering the dialog act recognition value and the relative number of words detected by the keyword spotter.

4.1 Sentence Templates

Each template has associated a set of conditions, e.g., the template is used only when a certain dialog act and topic occurs, or objects of certain types exist or have certain

properties. The templates themselves are a mixture of natural language words and hints where to place the content objects and how these objects should be generated. Additional markers can be inserted, allowing to discard portions of the template depending on the presence or absence of objects.

Generation hints have two tasks: first, the object has to be verbalized differently depending on the template, e.g., the `plane` object can be verbalized as *by plane* or *the flight*. Second, the number of needed templates is reduced, e.g., the `agent` object can produce *I could*, *could you* or *should we* depending on its content and given the hint `%could`. As a result, only one template must be defined instead of three using different conditions. The same reason motivated the introduction of markers. Since there is often more than one alternative in a template, the reduction effect is enormous. The sentence templates are transformed offline to FSTs, so the running system only requires an FST processor and no further formalism has to be supported.

4.2 Generation of Embedded Objects

After choosing the sentence pattern, the next step is to verbalize the embedded objects. We preferred direct FST coding for the verbalization of objects since templates aren't flexible enough to handle all the special cases caused by presence or absence of embedded objects.

For almost each object type an FST exists which is responsible for an adequate natural language generation of the object content and its embedded sub-objects. A dictionary is used for translating the representation to the target language, e.g., `central_station` to *Hauptbahnhof*. The Verbmobil LexDB is used to generate the determiners in German correctly, depending on number, case and gender of the noun.

4.3 An Example

The multi-lingual template that is selected for the example in section 3.1 is as follows:

```
Conditions: suggest traveling $book_action $move
English:    %should $agent book %for $ticket %without_prep
            $transportation &!> the journey &!< $*
German:    %sollen $agent %for $ticket %without_prep
            $transportation &!> die Fahrt &!< $* buchen
Japanese:  $* %without_prep $transportation &!> shutchou
            &!< no $ticket yoyaku shite %should_book $agent
```

Concepts realized in this example are: placeholders starting with `$`, generation hints starting with `%` which have effects like, e.g., `%for $tickets` generates to *two tickets for*, and markers starting with `&` which have effects like, e.g., *the journey* is only printed when no `transportation` object is available. The generated utterances from this template are:

English: *Should we book two tickets for the journey to Hamburg on the first.*
German: *Sollen wir zwei Tickets für die Reise nach Hamburg am ersten buchen.*
Japanese: *Shutchou no kippu o ni-mai yoyaku shite kudasaimasenka.*

5 Conclusion

In this contribution we gave a short overview of the dialog act based translation track of Verbmobil. It was designed as a robust module that is tolerant against defective input. It is similar to the approach as used in the C-Star system (Corazza, 1999). A recent evaluation of all translation tracks of Verbmobil shows that the dialog act based translation is competitive with the other translation tracks of Verbmobil.

As a side effect, we re-use the content representation as input for the dialog module. As demonstrated in the following articles on dialog processing (Chapter ??), and summary and minute generation (Chapter ??), this approach contributes significantly to the functionality of Verbmobil.

References

- Alexandersson, J., Buschbeck-Wolf, B., Fujinami, T., Kipp, M., Koch, S., Maier, E., Reithinger, N., Schmitz, B., and Siegel, M. (1998). Dialogue Acts in VERBMOBIL-2 – Second Edition. Verbmobil-Report 226, DFKI Saarbrücken, Universität Stuttgart, Technische Universität Berlin, Universität des Saarlandes.
- Appelt, D., Hobbs, J., Bear, J., and Tyson, M. (1993). FASTUS: A finite-state processor for information extraction from real-world text. In *Proceedings of the International Joint Conference on Artificial Intelligence*.
- Block, H. U. (1997). The Language Components in Verbmobil. In *Proceedings of International Conference on Acoustics, Signal and Speech Processing*, 79–82.
- Choi, W. S., Jeon-Mi-Cho, and Seo, J. (1999). Analysis System of Speech Acts and Discourse Structures Using Maximum Entropy Model. In *Proceedings of Association of Computational Linguistics*, 230–237.
- Corazza, A. (1999). An inter-domain portable approach to interchange format construction. In *Proceedings of EuroSpeech-99*, 2419–2426.
- Kipp, M. (1998). The Neural Path to Dialogue Acts. In *Proceedings of the European Conference on Artificial Intelligence*, 175–179.
- Klesen, M. (1997). Statistische Klassifikation von Dialogakten. Master's thesis, Universität des Saarlandes, Saarbrücken.
- Mast, M., Niemann, H., Nöth, E., and Schukat-Talamazzini, E. G. (1995). Automatic Classification of Dialog Acts with Semantic Classification Trees and Polygrams. International Joint Conference on Artificial Intelligence, Workshop on Machine Learning, Montreal.
- Reithinger, N., and Klesen, M. (1997). Dialogue Act Classification Using Language Models. In *Proceedings of EuroSpeech-97*, 2235–2238.
- Reithinger, N., Engel, R., Kipp, M., and Klesen, M. (1996). Predicting Dialogue Acts for a Speech-To-Speech Translation System. In *Proceedings of the International Conference on Speech and Language Processing*, 654–657.
- Samuel, K., Carberry, S., and Vijay-shanker, K. (1998). Computing Dialogue Acts from Features with Transformation-Based Learning. In *Proceedings of the American Association for Artificial Intelligence*, 90–97.

Tanaka, H., and Yokoo, A. (1999). An Efficient Statistical Speech Act Type Tagging System for a Speech Translation System. In *Proceedings of the Association for Computational Linguistics*, 381–388.