Chapter 10

Speech-to-Speech Translation

Stephan Vogel, Tanja Schultz,
Alex Waibel, and Seichii Yamamoto

10.1 Introduction

Speech-to-speech translation is the task of translating speech input in one language into speech output in another language. This process consists of three basic steps: speech recognition, translation, and speech generation. In this chapter, the main focus is on the translation component and the challenges posed by spoken language input. Discussions of multilingual speech recognition and speech generation aspects can be found in Chapters 4 to 7.

A variety of approaches to speech-to-speech translation have been developed, including interlingua-based, example-based, statistical, and transfer approaches. The first section of this chapter compares interlingua-based and statistical implementations within the framework of the NESPOLE! system developed at Carnegie Mellon University. It discusses decoding strategies allowing for flexible phrase reordering, spoken language specific problems, such as the removal of disfluencies, and coupling of the speech recognition and translation components. The second section of this chapter focuses on example-based and transfer approaches and describes their realization within the spoken language translation system developed at ATR in Japan.

Translation of *speech* (as opposed to text) is greatly complicated by the fact that spontaneously spoken speech is ill-formed and errorful. Speech translation is compounding three sources of errors: errors introduced by the speaker (disfluencies, hesitations), errors of the recognizer, and errors of the translation system. In addition to the need for robust individual components (recognition, translation, synthesis), a successful speech-to-speech machine translation system (SSMT) must also be error-tolerant in its architecture. It must allow for and represent near-miss alternatives at each level of processing, statistically score the likelihood of each hypothesis, and prune and reject unpromising candidates as new knowledge and information is applied at each processing stage.

In order to provide the translation component with multiple hypotheses, the recognition output string is replaced by an output graph of possible near-miss word hypotheses (the "word-lattice"), and the subsequent translation module selects the most likely sentence during its attempt to find a translation. The translation module should also allow for missing or wrong words and provide multiple interpretation results in its attempt to produce an output sentence. Two schools of thought diverge on how to achieve that: a direct approach (here, statistical machine translation) and an Interlingua approach. Both approaches handle ambiguity stochastically and by retaining and evaluating competing alternatives. Both also take advantage of semantic and domain knowledge to subsequently reduce ambiguity and extract useful information from an otherwise ill-formed sentence.

10.1.1 Speech Translation Strategies

In the following sections, we present several speech translation strategies that we have explored at our laboratories. We describe and evaluate each method according to the following criteria:

- First and foremost, spoken language input and translation performance have to be considered.
- Cost of adding a new language: As the number of languages grows, the number of language pairs increases quadratically. Translation based on a language independent intermediate representation (**Interlingua**) alleviates this problem by requiring translation

into the Interlingua only. However, it requires designing the Interlingua and developing analysis and generation modules. More flexible and automatically scalable approaches must be found.

- Cost of development: To achieve good performance, good coverage of the many possible translations from one language to the other has to be achieved. Therefore, if translation models are (semi-)manually designed, each new language and each domain require additional development effort.
- Cost of data collection: Performance is dramatically affected by the amount of available data. The cost of collecting large parallel or tagged corpora (tree banks) increases with the number of languages and/or domains.
- Explicit representation of meaning: It is useful to be able to represent meaning for a system to produce paraphrases and/or to link to other applications in addition to the translator.

10.1.2 Language Portability

Speech translation has made significant advances over the last years with several high-visibility projects focusing on diverse languages in restricted domains (e.g., C-Star, NESPOLE!, Babylon). While speech recognition emerged to be rapidly adaptable to new languages in large domains, translation still suffers from the lack of both hand crafted grammars for Interlingua-based approaches and large parallel corpora for statistical approaches. Both facts prevent the efficient portability of speech translation systems to new languages and domains. We believe that these limits of language- and domain-portable conversational speech translation systems can be overcome by relying more radically on learning approaches using easy/cheap-to-gather data and by applying multiple layers of reduction and transformation to extract the desired content in another language. Therefore, as shown in Figure 10.1 we cascade several stochastic source-channel models that extract an underlying message from a corrupt observed output. The three models effectively translate: (1) speech into word lattices (ASR), (2) ill-formed fragments of word strings into a compact well-formed sentence (Clean), and (3) sentences in one language into sentences in another (MT).

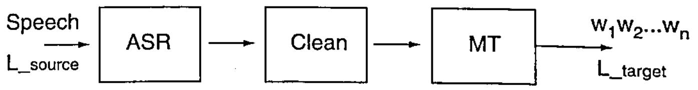


Figure 10.1: Stochastic source-channel speech translation system.

10.2 Statistical and Interlingua-Based Speech Translation Approaches

In response to these considerations, we begin by exploring direct and Interlingua-based translation approaches, as well as modifications on Interlingua-based approaches. While direct approaches can be trained on data, they require $O(N^2)$ parallel corpora if all of the N languages need to be connected. Interlingua-based approaches require fewer translation modules, but they typically require costly design and grammar development.

After a brief description of the Interlingua we introduce statistical machine translation methods. We then explore Interlingua-based statistical methods, interactively learning Interlingua and statistical translation via intermediary natural languages (pivot languages). Furthermore, we discuss the integration of speech recognition and translation modules into integrated speech translation systems. These integration methods are critical to scale speech translation systems to domain unlimited performance. Central aspects are cleaning disfluencies in speech and dealing with recognizer errors by translating not only the first best recognition result but all paths in the word lattice generated by the speech recognition system. Last but not least, we present a prototype of an integrated domain unlimited speech translation system.

10.2.1 Interlingua-Based Speech Translation

Interlingua-based MT analyzes a sentence into a language independent semantic representation using an analysis grammar, and generates the target sentence using a generation grammar. Therefore, building a domain-specific translation system first requires designing an Interlingua that is rich enough to capture the semantics of the domain. Second, analysis and generation grammars need to be developed.

Interlingua Design

The Interlingua, as for example used in the NESPOLE! system, is called **Interchange Format** (IF) (Levin et al., 1998). The IF defines a shallow semantic representation for task-oriented utterances that abstracts away from language-specific syntax and idiosyncrasies while capturing the meaning of the input. Each utterance is divided into semantic segments called semantic dialog units (SDUs), and an IF is assigned to each SDU. An IF representation consists of four parts: a speaker tag, a speech act, an optional sequence of concepts, and an optional set of arguments. The representation takes the following form:

speaker: speech act + concept* (argument*)

The speaker tag indicates the role of the speaker in the dialog, for example, "agent" and "client" in a hotel reservation dialog. The speech act captures the speaker's intention—for example, getting information or confirming. The concept sequence, which may contain zero or more concepts, captures the focus of an SDU—for example, getting information about the price of the hotel room. The speech act and concept sequence are collectively referred to as the domain action (DA). The arguments use a feature-value representation to encode specific information from the utterance (e.g., a double room, a nonsmoker). Argument values can be atomic or complex. Very simple and more complex examples of utterances with corresponding IFs are shown here:

On the twelfth we have a single and a double available.
a:give-information + availability + room
    (time = (md12), room-type = (single + double))

Thank you very much.
c:thank

The NESPOLE! Interlingua is based on representing the speaker's intention rather than the literal meaning of the utterance. The design of an Interlingua has to balance expressive power and simplicity. The inventory of domain actions needs to be sufficiently expressive in order to capture speaker intention. The specification at the argument level attempts to distinguish between domain-dependent and domain-independent sets of arguments, to better support portability to new domains. The Interlingua also has to be simple and straightforward enough so that grammar developers can independently work on different languages at different sites without the need of constant information exchange.

Semantic Grammars

The advantage of semantic grammars is that the parse tree that results from analyzing an utterance is very close to its final semantic interpretation (as opposed to, say, the laborious step of transforming syntactic constituent structures into semantic functional structures). A disadvantage however, is that a new grammar has to be developed for each domain, although some low-level modules—such as those covering time expressions—can be reused across domains. Once an utterance comes in, it is run through a parser that, together with the grammar in question, produces an analysis in the form of a parse tree. A statistical parser such as SOUP (Gavalda, 2004) is especially suitable for handling spoken language because it represents semantic grammars as probabilistic recursive transition networks. Thus, the search for the optimal interpretation is easily implemented as a beam search, with a scoring function that maximizes the coverage (number of words parsed) and likelihood (sum of arc probabilities) of an interpretation, but minimizes its complexity (number of nonterminals in parse lattice) and fragmentation (number of parse trees per utterance). Moreover, SOUP allows for skipping of words (with a penalty), character-level parsing (morphology rules can be defined using the same formalism as rules operating at the word level), multiple-tree interpretations (interpretations can be a sequence of top-level parse trees covering nonoverlapping fragments of the utterance), and dynamic modifications (rules can be modified on the fly), which enables the interactive grammar learning described below.

Interactively Learning Semantic Grammars

One of the greatest difficulties in the successful deployment of natural language processing (NLP) systems lies in the inherent richness of human language: it is impossible to capture a priori all the different ways in which people may choose to express a particular idea. Therefore, NLP

```
> do I have any mail
I understand "do I have any mail"
> arrange by recency
I don't understand right away what you mean but let me guess
"arrange by recency" is a way to express
 1. count mail, e.g., "count"
 2. list mail, e.g., "list"
 3. sort mail, e.g., "sort"
 0. None of the above
> sort
"recency" is a way to express
 1. sort by size, e.g., "size"
 2. sort by date, e.g., "date"
 3. sort by sender, e.g., "sender"
 0. None of the above
> by date
Thanks for teaching me the meaning of "arrange by recency"
I understand "arrange by recency"
> Please arrange messages from Bob by recency
I understand "please arrange messages from Bob by recency"
```

```
...
public <sortMail> = <_SORT>
   <_MAIL_ARGUMENT>  [<_SORT_MODE>]
    [<_SORT_BY>] [ <_SORT_BY> <_SORT_MODE>]:
...
<_SORT> = [please] (sort | arrange)

<sortBY_date> = date | time | recency:
...
```

Figure 10.2: Dynamic grammar acquisition as by-product of clarification dialogs using the GSG system on an e-mail client application. The left panel shows the dialog maintained between the user (preceded by ">") and the system; the right panel shows the dynamic extension of the underlying grammar, particularly the acquisition of the meaning of "arrange" and "recency."

systems that are capable of learning the meaning of extra-grammatical utterances—that is, capable of dynamically extending the coverage of their underlying grammars—become of paramount importance. An example is GSG (Gavalda, 2000), a system that extends the semantic grammar of a particular domain simply by asking clarification questions to the nonexpert user of the application in question (see Figure 10.2).

This is accomplished by augmenting the initial grammar using external knowledge sources (such as a part-of-speech tagger or a shallow syntactic grammar) in combination with various machine learning strategies within a coherent, mixed-initiative conversation with the end user. The result is the acquisition of new grammar rules. Then, a sophisticated rule management scheme—which includes detection of rule subsumption and rule ambiguity, vertical generalization (bottom-up generalization along ontological IS-A links), and horizontal generalization (making certain right-hand-side constituents optional and/or repeatable)—is employed to add the new rules to the current grammar, seamlessly and without disrupting analyses that are already correct.

10.2.2 Statistical Direct Translation

Statistical machine translation (SMT) was proposed in the early 1990s by the IBM research group (Brown et al., 1993b) and has since grown into a very active research field. Its key advantage is that it can be automatically trained from large corpora. The approach is based on Bayes' decision rule: Given a source sentence $\mathbf{f} = f_1^J$ of length J, the translation $\mathbf{e} = e_1^I$ is given by:

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}} p(\mathbf{e}|\mathbf{f}) = \arg\max_{\mathbf{e}} p(\mathbf{f}|\mathbf{e})p(\mathbf{e}). \tag{10.1}$$

Here, $p(\mathbf{e}) = p(e_1^I)$ is the language model of the target language, typically a trigram language model, and $p(f_1^J|e_1^I)$ is the **translation model**. The argmax operation denotes the search problem.

Word Alignment Models

A number of different translation models, also called alignment models, have been described (Brown et al., 1993b; Wu, 1995; Vogel et al., 1996; Och and Ney, 2000). The most simple **word alignment** models are based only on word co-occurrence statistics. In Brown et al. (1993b), this is the first in a sequence of models:

$$p(\mathbf{f}|\mathbf{e}) = \frac{1}{I^J}\prod_{j=1}^{J}\sum_{i=1}^{I} p(f_j|e_i). \tag{10.2}$$

This is the so-called **IBM1 model**. The **IBM2 model** also includes position-alignment probabilities $p(i|j,J,I)$, resulting in:

$$p(\mathbf{f}|\mathbf{e}) = \prod_{j=1}^{J}\sum_{i=1}^{I} p(f_j|e_i)p(i|j,J,I). \tag{10.3}$$

As an alternative to the IBM2 model, which is based on absolute positions, the so-called **HMM alignment model** has been formulated (Vogel et al., 1996), which uses relative positions to model word reordering between two languages:

$$p(\mathbf{f}|\mathbf{e}) = \sum_{a_1^J} \prod_{j=1}^{J} p(a_j|a_{j-1}, I) \cdot p(f_j|e_{a_j})$$

Here, a_j denotes the alignment to the j'th word in **f**.

Phrase Alignment Models

A simple approach to extract phrase translations from a bilingual corpus is to harvest the Viterbi path generated by a word alignment model. The phrase alignment is then essentially a post-processing step following the word alignment. For any word sequence in the source sentence, the aligned words in the target sentences are taken from the Viterbi alignment. The smallest and the largest index on the alignment side are then the boundaries for the entire target phrase aligned to the source phrase.

Many word alignment models are not symmetrical with respect to source and target language. To make up for the asymmetry of the word alignment models, training can be done in both directions: source to target, and target to source. This results in two Viterbi paths for each sentence pair. Different ways have been explored to combine the information given by those alignments. Och and Ney (2000) described experiments using intersection, union, and a combination along with some heuristic rules. Koehn et al. (2003) studied different combination schemes and concluded that using the right one has a bigger impact on the resulting performance of the translation system than the underlying word alignment model.

Some alternative phrase alignment approaches have been developed, which do not rely on the Viterbi word alignment. Both Marcu and Wong (2002) and Zhang et al. (2003b) consider a sentence pair as different realizations of a sequence of concepts. These alignment approaches segment the sentences into a sequence of phrases and align those phrases in an integrated way.

In Vogel et al. (2004) phrase alignment is described as sentence splitting, as shown in Figure 10.3. The goal is to find the boundaries for the

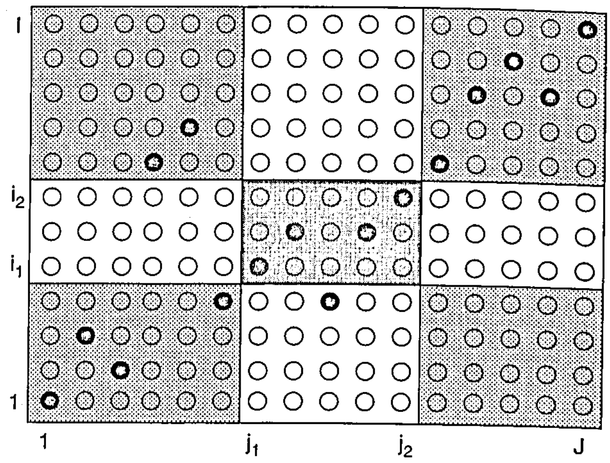


Figure 10.3: Phrase alignment as sentence splitting.

target phrase, given some source phrase. This can be done with a modified IBM1 alignment model:

- For words inside the source phrase, we sum only over the probabilities for words inside the target phrase candidate; for words outside of the source phrase, we sum only over the probabilities for the words outside the target phrase candidate.
- The position alignment probability—which for the standard IBM1 alignment is $1/I$, where I is the number of words in the target sentence—is modified to $1/(k)$ inside the source phrase and to $1/(I-k)$ outside the source phrase.

This leads to the following calculation:

$$p_{i_1,i_2}(\mathbf{f}|\mathbf{e}) = \prod_{j=1}^{j_1-1} \sum_{i \notin (i_1..i_2)} \frac{1}{I-k} p(f_j|e_i) \times \prod_{j=j_1}^{j_2} \sum_{i=i_1}^{i_2} \frac{1}{k} p(f_j|e_i) \times \prod_{j=j_2+1}^{J} \sum_{i \notin (i_1..i_2)} \frac{1}{I-k} p(f_j|e_i) \tag{10.4}$$

The target side boundaries i_1 and i_2, which give the highest alignment probability, are chosen by:

$$(i_1, i_2) = \arg\max_{i_1, i_2} \{p_{i_1,i_2}(\mathbf{f}|\mathbf{e})\}$$

Similar to $p_{i_1,i_2}(\mathbf{f}|\mathbf{e})$, we can calculate $p_{i_1,i_2}(\mathbf{e}|\mathbf{f})$—now summing over the source words and multiplying along the target words.

To find the optimal target phrase, we interpolate the log probabilities and take the pair (i_1, i_2) that gives the highest probability.

$$(i_1, i_2) = \arg\max_{i_1, i_2} \{(1-c)\log(\, p_{(i_1,i_2)}(\mathbf{f}|\mathbf{e})) + c \cdot \log(p_{(i_1,i_2)}(\mathbf{e}|\, \mathbf{f}))\}$$

The Decoder

A decoding approach that is based on

$$\hat{e} = \arg\max_{e} p(\mathbf{e}|\mathbf{f}) = \arg\max_{e} p(\mathbf{f}|\mathbf{e})p(e) \tag{10.5}$$

requires that, for any given word, its preceding word history is known. This leads to a search organization that constructs the target sentence in a sequential way. However, to incorporate the different word order of various languages, the words in the source sentence have to be covered nonsequentially while the translation is being generated. The most general form is to allow for any permutation of the source words—that is, without restrictions on the possible reordering. Such a search organization, restricted to word-based translation, has been described in Nießen et al. (1998). However, this leads to a very large search space and high computational costs. Therefore, various restrictions on reordering have been proposed.

The approach described here allows for phrase-to-phrase translation. Decoding proceeds along the source sentence. At each step, however, the next word or phrase to be translated may be selected from all words or phrases starting within a given look-ahead window from the current position. The decoding process works in two stages: First, the word-to-word and phrase-to-phrase translations and, if available, other specific information like named-entity translation tables are used to generate a translation lattice.

Second, a standard N-gram language model is applied to find the best path in this lattice. It is during this search step that reordering has to be taken into account. Both steps will now be described in more detail.

We define a transducer as a set of translation pairs generated by the methods described above as well as alternative knowledge sources, such as manual dictionaries and named entity lists. Each translation pair is given as a quadruple:

label # source words # target words # probability

For decoding, the transducers are organized as trees over the source side, and the translations are attached to the final nodes. This allows for efficient processing, since a node in the transducer represents all source phrases that consist of the words along the path to this particular node and include all possible paths that lead to final nodes of this particular node's subtree.

The first step in the decoding process is to build a translation lattice by applying the transducers. We convert the sentence to be translated into a lattice structure, in which the nodes are the positions between the words, and the words are attached to the edges. The nodes ν are numbered from 0 to J, the length of the source sentence. We also use ν to simply denote the node number.

To search for matching phrases, we encode the relevant information in a hypothesis structure

$$h = (\nu_1, \nu_2, \sigma, h_p, \varepsilon),$$

which means that starting from node ν_1 and ending in node ν_2, a sequence of words has been found that corresponds to a transducer path from state σ_0 to state σ, and whereby in the last step, the hypothesis h_p has been expanded over edge ε.

The matching process between a path through a transducer and a segment of a sentence can start at all positions in the sentence. Therefore, an initial hypothesis $(\nu, \nu, \sigma = \sigma_0, h_p = \emptyset, \varepsilon = \emptyset)$ is set for each node except the final node in the lattice.

Expanding hypotheses is structured in the following way: Let ν be a node in the translation graph, and $E(\nu)$ be the set of incoming edges for

this node. Let $\nu^s(\varepsilon)$ denote the start node of an edge ε. Then, for each incoming edge $\varepsilon \in E(\nu)$, all hypotheses h_p in $\nu^s(\varepsilon)$ are expanded with the word f attached to ε. That is to say, if σ_p is the transducer state of hypothesis h_p, then σ is the transducer state that can be reached from σ_p over the transition labeled f. If expansion is possible, then a new hypothesis h is generated:

$$h_p = (\nu_1, \nu^s(\varepsilon), \sigma_p, h', \varepsilon') \rightarrow h = (\nu_1, \nu, \sigma, h_p, \varepsilon).$$

If expanding a hypothesis leads into a final state of the transducer, a new edge is created and added to the translation lattice. The new edge is labeled with the category label taken from the transducer. The additional information stored with this edge is the translation and the sequence of edges traversed, which corresponds to the sequence of source words.

Once the complete translation lattice has been built, a one-best search through this lattice is performed. In addition to the translation probabilities, or rather translation costs, as we use the negative logarithms of the probabilities for numerical stability, the language model costs are added and the path that minimizes the combined cost is returned.

The search for the best translation hypothesis involves generating partial translations and expanding them until the entire source sentence has been accounted for. The information accumulated during search is stored in the following hypothesis structure:

$$h = (Q, C, \Lambda, i, h_p, \varepsilon),$$

where Q is the total cost; C, the coverage information; Λ, the language model state; i, the number of the words in the partial translation; and h_p and ε, the trace-back information. In the case of a trigram language model, the language model state comprises just the last two words of the partial translation—that is, the history in the next expansion step.

To allow for reordering, we organize the search in the following way. Assume we have a partial translation, which already covers c words of the source sentence, $n < c$ of which are the first words of the sentence. (In other words, the initial section of the sentence has already been completely translated, the remainder only partially.) To expand this partial translation, we have to extend it over one of the edges in the translation

Second, a standard N-gram language model is applied to find the best path in this lattice. It is during this search step that reordering has to be taken into account. Both steps will now be described in more detail.

We define a transducer as a set of translation pairs generated by the methods described above as well as alternative knowledge sources, such as manual dictionaries and named entity lists. Each translation pair is given as a quadruple:

label # source words # target words # probability

For decoding, the transducers are organized as trees over the source side, and the translations are attached to the final nodes. This allows for efficient processing, since a node in the transducer represents all source phrases that consist of the words along the path to this particular node and include all possible paths that lead to final nodes of this particular node's subtree.

The first step in the decoding process is to build a translation lattice by applying the transducers. We convert the sentence to be translated into a lattice structure, in which the nodes are the positions between the words, and the words are attached to the edges. The nodes ν are numbered from 0 to J, the length of the source sentence. We also use ν to simply denote the node number.

To search for matching phrases, we encode the relevant information in a hypothesis structure

$$h = (\nu_1, \nu_2, \sigma, h_p, \varepsilon),$$

which means that starting from node ν_1 and ending in node ν_2, a sequence of words has been found that corresponds to a transducer path from state σ_0 to state σ, and whereby in the last step, the hypothesis h_p has been expanded over edge ε.

The matching process between a path through a transducer and a segment of a sentence can start at all positions in the sentence. Therefore, an initial hypothesis $(\nu, \nu, \sigma = \sigma_0, h_p = \emptyset, \varepsilon = \emptyset)$ is set for each node except the final node in the lattice.

Expanding hypotheses is structured in the following way: Let ν be a node in the translation graph, and $E(\nu)$ be the set of incoming edges for

this node. Let $\nu^s(\varepsilon)$ denote the start node of an edge ε. Then, for each incoming edge $\varepsilon \in E(\nu)$, all hypotheses h_p in $\nu^s(\varepsilon)$ are expanded with the word f attached to ε. That is to say, if σ_p is the transducer state of hypothesis h_p, then σ is the transducer state that can be reached from σ_p over the transition labeled f. If expansion is possible, then a new hypothesis h is generated:

$$h_p = (\nu_1, \nu^s(\varepsilon), \sigma_p, h', \varepsilon') \rightarrow h = (\nu_1, \nu, \sigma, h_p, \varepsilon).$$

If expanding a hypothesis leads into a final state of the transducer, a new edge is created and added to the translation lattice. The new edge is labeled with the category label taken from the transducer. The additional information stored with this edge is the translation and the sequence of edges traversed, which corresponds to the sequence of source words.

Once the complete translation lattice has been built, a one-best search through this lattice is performed. In addition to the translation probabilities, or rather translation costs, as we use the negative logarithms of the probabilities for numerical stability, the language model costs are added and the path that minimizes the combined cost is returned.

The search for the best translation hypothesis involves generating partial translations and expanding them until the entire source sentence has been accounted for. The information accumulated during search is stored in the following hypothesis structure:

$$h = (Q, C, \Lambda, i, h_p, \varepsilon),$$

where Q is the total cost; C, the coverage information; Λ, the language model state; i, the number of the words in the partial translation; and h_p and ε, the trace-back information. In the case of a trigram language model, the language model state comprises just the last two words of the partial translation—that is, the history in the next expansion step.

To allow for reordering, we organize the search in the following way. Assume we have a partial translation, which already covers c words of the source sentence, $n < c$ of which are the first words of the sentence. (In other words, the initial section of the sentence has already been completely translated, the remainder only partially.) To expand this partial translation, we have to extend it over one of the edges in the translation

Second, a standard N-gram language model is applied to find the best path in this lattice. It is during this search step that reordering has to be taken into account. Both steps will now be described in more detail.

We define a transducer as a set of translation pairs generated by the methods described above as well as alternative knowledge sources, such as manual dictionaries and named entity lists. Each translation pair is given as a quadruple:

label # source words # target words # probability

For decoding, the transducers are organized as trees over the source side, and the translations are attached to the final nodes. This allows for efficient processing, since a node in the transducer represents all source phrases that consist of the words along the path to this particular node and include all possible paths that lead to final nodes of this particular node's subtree.

The first step in the decoding process is to build a translation lattice by applying the transducers. We convert the sentence to be translated into a lattice structure, in which the nodes are the positions between the words, and the words are attached to the edges. The nodes ν are numbered from 0 to J, the length of the source sentence. We also use ν to simply denote the node number.

To search for matching phrases, we encode the relevant information in a hypothesis structure

$$h = (\nu_1, \nu_2, \sigma, h_p, \varepsilon),$$

which means that starting from node ν_1 and ending in node ν_2, a sequence of words has been found that corresponds to a transducer path from state σ_0 to state σ, and whereby in the last step, the hypothesis h_p has been expanded over edge ε.

The matching process between a path through a transducer and a segment of a sentence can start at all positions in the sentence. Therefore, an initial hypothesis $(\nu, \nu, \sigma = \sigma_0, h_p = \emptyset, \varepsilon = \emptyset)$ is set for each node except the final node in the lattice.

Expanding hypotheses is structured in the following way: Let ν be a node in the translation graph, and $E(\nu)$ be the set of incoming edges for

this node. Let $\nu^s(\varepsilon)$ denote the start node of an edge ε. Then, for each incoming edge $\varepsilon \in E(\nu)$, all hypotheses h_p in $\nu^s(\varepsilon)$ are expanded with the word f attached to ε. That is to say, if σ_p is the transducer state of hypothesis h_p, then σ is the transducer state that can be reached from σ_p over the transition labeled f. If expansion is possible, then a new hypothesis h is generated:

$$h_p = (\nu_1, \nu^s(\varepsilon), \sigma_p, h', \varepsilon') \rightarrow h = (\nu_1, \nu, \sigma, h_p, \varepsilon).$$

If expanding a hypothesis leads into a final state of the transducer, a new edge is created and added to the translation lattice. The new edge is labeled with the category label taken from the transducer. The additional information stored with this edge is the translation and the sequence of edges traversed, which corresponds to the sequence of source words.

Once the complete translation lattice has been built, a one-best search through this lattice is performed. In addition to the translation probabilities, or rather translation costs, as we use the negative logarithms of the probabilities for numerical stability, the language model costs are added and the path that minimizes the combined cost is returned.

The search for the best translation hypothesis involves generating partial translations and expanding them until the entire source sentence has been accounted for. The information accumulated during search is stored in the following hypothesis structure:

$$h = (Q, C, \Lambda, i, h_p, \varepsilon),$$

where Q is the total cost; C, the coverage information; Λ, the language model state; i, the number of the words in the partial translation; and h_p and ε, the trace-back information. In the case of a trigram language model, the language model state comprises just the last two words of the partial translation—that is, the history in the next expansion step.

To allow for reordering, we organize the search in the following way. Assume we have a partial translation, which already covers c words of the source sentence, $n < c$ of which are the first words of the sentence. (In other words, the initial section of the sentence has already been completely translated, the remainder only partially.) To expand this partial translation, we have to extend it over one of the edges in the translation

lattice that corresponds to one of the remaining untranslated source words. We allow for phrases—that is, longer edges in the translation lattice. It can be the case, therefore, that such an edge spans over some words that have already been covered. This causes a collision, and so an expansion over such an edge is not possible.

Reordering is now restricted to be within a window of given size. That is to say that the next word to be translated has to be taken from positions $n <= j <= n + d$, where d is the size of the reordering window. In terms of nodes: if ν_1 is the node with number n and ν_2 is the node with number $n + d$, then expansion is restricted to edges starting from nodes $\nu_1 <= \nu' <= \nu_2$. With $d = 0$, there is no reordering; therefore, decoding is monotone.

Expansion of hypotheses is organized according to overall coverage—that is, the number of words already translated. So we start with coverage zero and expand until we have reached coverage J, where J is the number of words in the source sentence. At the sentence end, the language model probability for the sentence end is applied. In addition, a sentence-length model can be used. The best hypothesis is then used to trace back and collect the actual words generated along this path.

To reconstruct the path taken through the translation lattice, we need to store additional back-pointer information. Traveling back using these pointers allows us to generate the actual sequence of words. The back-pointer information consists of the edge that was traversed during the last expansion and the pointer to the predecessor of the current hypothesis.

10.2.3 Statistical Translation Using a Formal Interlingua

Interlingua-based translation as described above requires, in addition to the design of the interlingua, the development of handwritten (or interactively learned) semantic grammars—analysis grammars for each input language and generation grammars for each output language. Here we describe a method to automatically train a semantic mapping between source text and the tree-structured interlingua, which replaces the analysis grammar. We show that this can be done given a corpus of semantically tagged data (from source language to IF).

A Language Model for Trees

In the usual situation, where $\mathbf{e} = (e_1, \ldots, e_l)$, language modeling is typically based on the decomposition

$$p(\mathbf{e}) = \prod_{i=1}^{l} p(e_i | e_1, \ldots, e_{i-1}),$$

where the conditional probability $p(e_i | e_1, \ldots, e_{i-1})$ is approximated by the relative frequencies of N-grams seen in the training corpus. While $\mathbf{e}$ may in this case be defined as some token e together with a subsequence $\mathbf{e}'$, a tree $\mathbf{e}$ may be defined as consisting of some token e together with a set of $a \geq 0$ subtrees $\mathbf{e}_1, \ldots, \mathbf{e}_a$ (a is the arity of the tree). This leads to the decomposition

$$p(\mathbf{e}) = p(e | \mathbf{e}_1, \ldots, \mathbf{e}_a) \cdot \prod_{i=1}^{a} p(\mathbf{e}_i | \mathbf{e}_1, \ldots, \mathbf{e}_{i-1}),$$

which corresponds to a bottom-up decoding in the order $\mathbf{e}_1, \ldots, \mathbf{e}_a, e$.

It is a special feature of the IF that the ordering of subtrees is unimportant for the semantics they cover—that is, the term $a(b, c)$ is semantically equivalent to $a(c, b)$. This justifies the assumption that the probabilities $p(\mathbf{e}_i | \mathbf{e}_1, \ldots, \mathbf{e}_{i-1})$ are independent of $\mathbf{e}_1, \ldots, \mathbf{e}_{i-1}$, given the recursive formula

$$p(\mathbf{e}) = p(e | \mathbf{e}_1, \ldots, \mathbf{e}_a) \cdot \prod_{i=1}^{a} p(\mathbf{e}_i),$$

in which $p(\mathbf{e}_i)$ is to be decomposed further in the same way as $p(\mathbf{e})$. To approximate $p(e | \mathbf{e}_1, \ldots, \mathbf{e}_a)$ with relative frequencies, "tree-N-grams" are used. As Figure 10.4 shows, these N-grams use only the roots of the subtrees.

Translation Models for Trees

As described previously, the standard translation models use the concept of word alignment: each word in the source sentence is aligned to a word

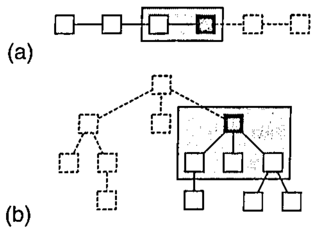


Figure 10.4: The concept of N-grams (a) in sequences (b) in trees.

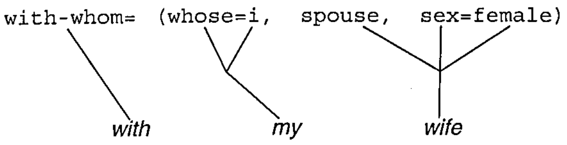


Figure 10.5: An alignment between an English phrase and its corresponding IF representation.

in the target language. Words that have no correspondence in the target sentence are aligned to the so-called empty word added to the sentence at position 0. This concept of alignment can also be used when translating into IF, as illustrated in Figure 10.5.

The IBM1 and IBM2 translation models proposed in Brown et al. (1993b) can be generalized to the case in which the e's are trees. For the IBM1 translation model, this is straightforward because the model makes no assumptions that are specific to sequences. In the model's formula

$$p(\mathbf{f}|\mathbf{e}) = \frac{1}{(I+1)^J} \prod_{j=1}^{J} \sum_{i=0}^{I} p(f_j|e_i),$$

we just need to assign an index to each of the I nodes in the tree $\mathbf{e}$ in some arbitrary way.

The IBM2 model also includes alignment probabilities $p(i|j,I,J)$, resulting in the estimation formula

$$p(\mathbf{f}|\mathbf{e}) = \prod_{j=1}^{J} \sum_{i=0}^{I} p(f_j|e_i)p(i|j,J,I)$$

in the sequential case. The idea is that the jth token of the source sentence **f** is aligned to the ith token of the target sentence **e** with probability $p(i|j,J,I)$ provided that J is the length of **f** and I is the length of **e**.

Now, indexing the nodes of a tree is no longer arbitrary because it affects the values of $p(i|j,J,I)$. Using again the fact that the IF is commutative, it seems appropriate to consider only the depth of a particular node. Its position within the level does not contribute any information. Taking d as the depth of **e** and $\#e_i$ as the number of nodes in level i, this leads to

$$p(\mathbf{f}|\mathbf{e}) = \prod_{j=1}^{J} \sum_{i=0}^{d} \left(\frac{p(i|j,J,d)}{\#e_i} \sum_{e_i} p(f_j|e_i)\right)$$

where the e_i in the right sum run over all nodes of level i.

Decoding Trees

For generating **e** from **f** given a language model $p(\mathbf{e})$ as well as a translation model $p(\mathbf{f}|\mathbf{e})$, a stack decoder similar to the one described in Wang and Waibel (1997) is used but adapted to generate trees rather than linear sequences. To cope with the huge search space, "bad" hypotheses are pruned after each iteration.

The algorithm starts with an empty hypothesis. In the case of linear sequences, new hypotheses are generated by iteratively appending new target words to existing hypotheses. The tree decoder takes a *set* of existing hypotheses $\mathbf{h}_1, \ldots, \mathbf{h}_n$ and forms a new tree, with the $\mathbf{h}_i$ as subtrees and an additional target token as its root. If the algorithm is restricted to choose only sets of size 1, it reduces to the sequential version.

Generating trees gives a much larger search space than generating linear sequences. In fact, while n hypotheses and k words to append lead to $n \cdot k$

new hypotheses for the next iteration of the sequence decoder, the tree decoder generates $2^n \cdot k$ new hypotheses because a set of size n has exactly 2^n subsets.

Three methods are applied to reduce the search space. First, hypotheses that are not legal terms according to the IF specification are not generated. Second, the branching factor of generated trees is restricted to three, the depth to four. Finally, standard pruning is used. In less than 5% of the test sentences used in our experiments, the decoder generated an IF that had a lower score than the reference IF, indicating that the number of search errors due to pruning was small.

10.2.4 Using English as Interlingua

Interlingua-based translation systems and statistical translation systems are both well-known approaches with inherent advantages. However, in most systems, only one of the approaches can be fully implemented, at the expense of the other. Using English as interlingua tries to combine the advantages of a system with an explicit Interlingua and the advantages of a pure data-driven system. The Error-Driven Translation Rule Learning (EDTRL) system overcomes existing limitations by (1) avoiding the need for an explicit handcrafted interlingua specification and (2) tackling the "Parallel Data Sparseness Problem" that statistical machine translation faces for unusual or low-resources language pairs.

While translation from and to English has made significant progress partly due to large parallel corpora, the situation drastically changes if non-English language pairs are supposed to be translated into each other. The amount of parallel text corpora is much smaller than the parallel text corpora from each of these languages paired with English. The intuitive solution to this problem is to cascade two translators using English as an intermediate language. However, the pure cascading of two machine translation systems using the output of the first as input to the second results in a multiplication of translation errors and therefore in a significantly higher error rate compared to each translation step. The reduction of this multiplication in translation errors should be achieved by introducing a suitable interlingua and appropriate training and decoding methods. This is the focus of the EDTRL approach.

Standardized and Simplified English as Interlingua

Using ordinary English makes it more difficult to take advantage of formal aspects of an interlingua. To take this into account and to improve the cascaded translation, the intermediate English is transformed to a standardized and simplified form. The standardization step maps alternative expressions with similar or equal meanings to the most commonly used alternative. Sometimes English utterances have some freedom in word order without changing the main meaning of the utterance. To obtain a consistent word order in such cases, a reordering step can be applied, as in the following:

please give me ... → give me ... please

Furthermore, the sentence structure is simplified (SimplifiedEnglish, 2004), with more complex, rarely used tenses being replaced by easier ones:

He had spoken. → He spoke.
He would be speaking. → He would speak.

Although these kinds of simplifications do remove information, often such fine nuances are of little value to the quality of the translation given the current performance of MT systems. In most cases, the translation profits from the transformations through more reliable alignments and better utilization of the training data. Even humans can benefit from Simplified English in some technical domains (AECMA, 2004).

Linguistically Enhanced English as Interlingua

Besides standardizing and simplifying the intermediate English, adding further information to the structure and the semantic content of a sentence can be helpful for the second translation step. To be independent of the source and target language and to minimize manual work, additional knowledge sources should only use information that can be automatically obtained from parallel text or be derived from the intermediate representation based on English as interlingua.

We examined the incorporation of the following additional knowledge sources to provide additional information for the translation process:

- Morphological Analyzer: Starting from the WordNet ontology (Miller et al., 2005), we built a system to analyze English word forms and determine its base forms and derivation rules. The analyzer contains a set of common transformation rules and an even larger list of exceptions from these rules. In the current implementation, each word is analyzed without using its context or information from former sentences. The precision for finding the base class is 95%, while the determination of the derivation rules is not yet that good.
- Sense Guesser: The sense guesser tries to find the sense of a word. Many words have different meanings depending on the context in which they occur—for example, "table" can have the senses "desk" or "chart." Often the context of the word can be used for disambiguation. In our example, the context "in the" assigns "table" to the chart class, while "on the" assigns it to the desk class. We used the sense hierarchy from WordNet.
- Synonym Generator: WordNet also lists synonyms for words, all within a well-structured and linked hierarchy. Both the sense guesser and synonym generator only use open word classes like nouns, verbs, adjectives, and adverbs.
- Part-of-Speech Tagger: A statistical part-of-speech tagger is used to provide POS-tags. The tagger uses the tag set described in Brill (1995) and trained on the tagged Brown corpus.
- Named Entity Tagger: Handwritten rules are used to find named entities, which often need to be treated in a special way.

Further knowledge sources such as sentence type, active or passive voice, politeness, domain, and category could also be added.

Connecting the Translation Steps

The translation errors from intermediate English to the target language can be reduced if not only the best hypothesis but also additional information from the search is used. We have examined the following methods:

- N-best list of complete translations: The translation system produces up to N alternative translation hypotheses and passes them to the second translation step. The number of hypotheses has to be kept small to guarantee fast overall decoding, thereby allowing only for little variability. This approach did not improve the translation in our experiments.

- N-best word or phrase alternatives to the best hypothesis: This method selects the single best hypothesis from the first translation step, but augments it by adding alternative words or phrases that have high translation probabilities. This strategy results in a noticeable improvement in the translation performance.
- Full lattice: In order not to fix one translation hypothesis as the basis for constructing these alternatives, we can also pass on full translation lattices. This method has the highest potential because it keeps all promising alternatives. But without pruning, this approach increases the search space considerably. Using a lattice as input for the second translation step has been shown as the most profitable way to use translation alternatives to improve the translation quality.

Besides the information about alternatives and additional knowledge sources their probability or confidence measure can be part of the interlingua. Therefore words, phrases, and their alternatives carry probabilities as well as attributes and classes. All this information together with the formalization step forms the interlingua, which allows improvement of the cascaded translation.

10.2.5 Comparing the Translation Strategies

In this section, we evaluate the different translation strategies on two speech-to-speech translation tasks. In the first part, we compare the grammar-based system to the statistical interlingua-based system and to the direct statistical approach. (The experiments were carried out using travel planning dialogs from the NESPOLE! project [Lavie et al., 2001a].) In the second part, a comparison between the Error-Driven Translation Rule Learning (EDTRL) system and the direct statistical system on the BTEC (Kikui et al., 2003) corpus is presented.

Interlingua versus Direct Statistical Translation

Dialogs in the travel planning domain have been collected, transcribed, and annotated with IF representations (Lavie et al., 2001a). From this

Table 10.1 Corpus statistics for the NESPOLE! training and test set.

	Training		Test	
Language	Ger	Eng	Ger	Eng
Sentences	2,427	2,427	194	194
Tokens	11,236	11,729	889	955
Vocabulary	1,196	1,010	269	241
Singletons	566	429	152	123

Table 10.2 Scores of the translations generated by systems *IL*, *SIL*, and *SMT*; the values are percentages and averages of four independent graders.

	IL	*SIL*	*SMT*
Perfect	18.9	15.1	40.3
Okay	36.3	30.2	22.7
Bad	44.8	54.7	37.0
Acceptable	**55.2**	**45.3**	**63.0**

database, we extracted a trilingual corpus of about 2,500 triples German-English-IF as a training set. One hundred ninety-four German sentences were held out to be used as a test set. Detailed corpus statistics are given in Table 10.1.

For each German test sentence, three IF representations were generated using (1) the grammar-based system (*IL*); (2) the statistical system (*SIL*), with a model trained on German/IF; and (3) the direct statistical translation system (*SMT*). For the interlingua-based systems, the IF expressions were converted into English using the same IF-to-English generation grammar.

The translations from the different systems were then presented to six human evaluators. Each translation was assigned one of three grades: "perfect" (the translation is semantically complete and grammatically correct); "okay" (the main part of the original semantics is covered and expressed understandably); "bad" (otherwise). The "perfect" and "okay" translations form the class "acceptable."

The evaluation results are given in Table 10.2. The statistical IL system does not perform quite as well as the grammar-based system. Given the very small training corpus, with about 40% of all words seen only once during training, this is not surprising, on the contrary, the results show the potential of the proposed approach. However, the direct statistical system—not using any syntactic structure information beyond what is implicit in the phrase-to-phrase alignments—outperformed the grammar-based system, despite the small training corpus and the significant amount of effort that had been put into the development of the grammars.

EDTRL versus Direct Statistical Translation

In the following experiments, we first evaluate the concept of English as an interlingua, and then compare this to the direct statistical translation.

To evaluate the concept of English as an interlingua, we chose Chinese as the input language and Spanish as the output language, since, in spite of the widespread use of these languages, comparatively few direct Chinese-Spanish translations are available. We trained the EDTRL system for Chinese to English ($C \rightarrow E$), English to Spanish ($E \rightarrow S$), and Chinese to Spanish ($C \rightarrow S$). We then cascaded the $C \rightarrow E$ and $E \rightarrow S$ systems by simply feeding the output of the former into the latter. The translation was then done on the same test set using the full definition of an augmented, formalized version of English as an intermediate step. Additionally, we trained a statistical MT system on the same language pairs and cascaded the $C \rightarrow E$ and $E \rightarrow S$ translations to generate a $C \rightarrow E \rightarrow S$ translation in comparison to a direct $C \rightarrow S$ translation. To evaluate the translation quality, we used the NIST standardized tool for benchmark evaluations (MTeval) in version 10 (NIST, 2000). For comparison, we give also the results for Systran's publicly available online machine translation system.

The SMT system and the EDTRL system both use the same bilingual training corpus, while the EDTRL system uses additional dictionaries for initialization. An additional difference is the handling of punctuation. While EDTRL ignores punctuation marks, SMT treats them as normal words. In the reported experiments, the EDTRL system does not make use of the sense guesser, the named entity tagger, and full lattice. The data for these experiments were taken from the Basic Travel Expression Corpus (BTEC), a multilingual collection of conversational phrases in the travel domain (Kikui et al., 2003) as briefly introduced in Section 10.4.2. Table 10.3 shows the training and test material for Chinese, English, and Spanish phrases. Since only a subset of 6,027 phrases was available for

Table 10.3 Training (test in parentheses) corpora.

Train (Test)	Chinese	English	Spanish
sentences	162,316 (506)	162,316 (506)	6,027
-unique	96,074 (497)	97,500 (503)	5,934
-avg. length	7.0 (7.3)	7.5 (7.5)	9.8
words	1,134,417 (3681)	1,216,207 (3779)	58,834
vocabulary	13,793 (954)	16,224 (843)	4,651
-singletons	4,745 (590)	6,705 (523)	2,370
-unseen	(29)	(22)	

Spanish, the training data for the E → S and C → S systems was reduced to the corresponding parallel phrases. The scores were calculated based on 16 English and 3–4 Spanish reference translations.

The first four lines in Table 10.4 give NIST scores of direct translations from the source to the target language. For the direct translation, the EDTRL system does not use any interlingua. While the lower rows refer to an internal evaluation of a preliminary version of the systems (01/2004), both systems were lately compared among several systems in an official evaluation (IWSLT, Kyoto Japan, August 2004). Based on the NIST score of the unlimited Chinese-English track, the SMT system came in first and the EDTRL system came in second.

The second and third column of Table 10.4 show the comparison between EDTRL and direct statistical translation. As the fifth line shows, the cascaded EDTRL system outperforms the directly trained systems. Using augmented and formalized English as interlingua yields to further improvements of the EDTRL system (last line of Table 10.4).

Table 10.4 NIST scores for translation from Chinese to Spanish.

Translation tasks	EDTRL	SMT	Systran
C → E (IWSLT eval 08/2005)	7.5	9.56	-
C → E (internal eval 01/2004)	7.34	7.35	5.74
E → S	5.17	4.57	6.06
C → S	3.17	3.04	-
C → E → S	3.41	2.60	2.84
C → E_{IL} → S	3.69	-	-

10.3 Coupling Speech Recognition and Translation

Due to the peculiarities of spoken language, an effective solution to speech translation cannot be expected to be a mere sequential connection of automatic speech recognition (ASR) and machine translation components but rather a coupling between both. This coupling can be characterized by three orthogonal dimensions: (1) the complexity of the search algorithm, (2) the incrementality, and (3) the tightness, which describes how close ASR and MT interact while searching for a solution (Ringger, 1995). The benefits and drawbacks have been widely discussed along aspects such as modularity, scalability, and complexity of systems (Ringger, 1995; Harper et al., 1994). State-of-the-art translation systems use a variety of different coupling strategies. Examples of loosely coupled systems are IBM's MASTOR (Liu et al., 2003), ATR-MATRIX (Takezawa et al., 1998c), and NESPOLE! (Lavie et al., 2001a), which uses the interlingua-based JANUS system. Examples for tightly coupled systems are EuTrans (Pastor et al., 2001), developed at UPV, and AT&T's Transnizer (Mohri and Riley, 1997).

10.3.1 Removing Disfluencies

Spontaneous spoken speech usually contains disfluencies such as filler words, repairs, or restarts, which do not contribute to the meaning of the spoken utterance and cause sentences to be ill-formed, longer, and thus harder to process for translation. We developed a cleaning component based on a noisy-channel model that automatically removes these disfluencies (Honal and Schultz, 2003, 2005). Its development requires no linguistic knowledge but rather annotated texts and therefore has large potential for rapid deployment and adaptation to new languages.

In this approach, we assume that "clean" (i.e., fluent) speech gets passed through a noisy channel that adds "noise" to the clean speech, and thus outputs disfluent speech. Given a noisy string N, the goal is to recover the clean string C such that $p(C|N)$ becomes maximal. Using Bayes' rule, this problem can be expressed as:

$$\hat{C} = \arg\max_C P(C|N) = \arg\max_C P(N|C) \cdot P(C). \qquad (10.6)$$

We model the probability $P(C)$ with a trigram language model trained on fluent speech. To establish correspondences between the positions of the source and the target sentences, word-alignment models as described previously can be used. However, in the case of disfluency cleaning, only deletions of words needs to be considered. Assuming that each target sentence is generated from left to right, the alignment a_j defines whether the word n_j in the source sentence is deleted or appended to the target sentence. Let J be the length and n_j the words of the source sentence N, I the length, and c_i the words of the target sentence C; and m the number of deletions (of contiguous word sequences) that are made during generation of the target sentence. We can then introduce an alignment a_j for each word n_j and rewrite $P(N|C)$ as:

$$P(N|C) = P_{I,J}(m) \cdot \prod_{j=1}^{J} P_w(n_j). \tag{10.7}$$

The probability $P_{I,J}(m)$ models the number m of contiguous word sequences that can be deleted in N to obtain C. $P_w(n_j)$ is the probability that word n_j of the string N is disfluent.

Each of the probabilities $P_w(n_j)$ is finally composed of a weighted sum over the following six models: (M1) models the length of the deletion region of a disfluency; (M2) models the position of a disfluency; (M3) models the length of the deletion region of a disfluency with a word fragment at the end of the reparandum; (M4) models the context of a potentially disfluent word; (M5) uses information about the deletions of the last two words preceding a potentially disfluent word; and (M6) takes into account whether a potentially disfluent word is part of a repeated word sequence.

The system can be optimized on a development test set by training the scaling factors for the different models using a gradient descent approach.

The probability distributions for the models are obtained from the training data using relative frequencies. All experiments are conducted on spontaneously spoken dialogs in English from the Verbmobil corpus, and, in order to demonstrate the feasibility of rapid adaptation, on the spontaneous Mandarin Chinese CallHome corpus. The highest performance gain results from model (M4), which considers the context of a potentially disfluent word. This can be easily explained for filler words, since it allows discriminating between the deletion of the word "well" in the

Table 10.5 Results for automatic disfluency removal on the English Verbmobil (EVM) and the Chinese CallHome (CCH) corpora.

Corpus	Setup	Precision	Recall	F_1
EVM	Baseline	90.2	77.2	0.832
	Hand optimized	91.5	86.2	0.888
	Gradient descent	93.1	85.1	0.890
CCH	Baseline	76.8	49.4	0.601
	Hand optimized	77.8	53.4	0.634
	Gradient descent	79.0	53.0	0.634

context “Well done!” and “Alright, well, this is a good idea.” The impact of (M1), (M2), and (M3) is a slight increase of the number of hits at the cost of a slight increase or decrease of the number of false positives. Model (M5) causes a large number of false positives and was therefore disregarded in the best system.

Overall, the baseline system shows a precision of 90.2% and a recall of 77.2% for English dialogs, as shown in Table 10.5. Almost no effort was required for the adaptation to Mandarin Chinese. The same algorithms and the same statistical models were used, achieving 76.8% precision and 49.4% recall on the Mandarin corpus (after retraining on the CallHome data). When adjusting the weighting parameters for the models, a small improvement was achieved.

10.3.2 Lattice Coupling

Research on spoken language translation must support scalable systems capable of handling complex translation tasks, but it must also allow for the improvement of individual components. Our own system is therefore structured in a loosely coupled, nonincremental way. Initially, the link between the ASR and MT components was established solely through the single best hypothesis generated by the speech recognizer. Since it is well known that the speech recognition word error rate can be dramatically decreased by generating many alternatives in the form of N-best lists or word lattices, we expect that it could also improve translation performance.

We conducted several studies to investigate whether lattice-based coupling between ASR and MT improves speech translation performance.

The studies were carried out on German-to-English travel-arrangement dialogs originally recorded as part of the NESPOLE! speech translation project (Lavie et al., 2001a). The applied version of the JANUS speech recognition toolkit (JRTk) (Metze et al., 2003) achieved 23.5% lattice word error rate in 1.3× real time, given a lattice density of 22. In JRTk, a word lattice is represented as a directed graph in which nodes are associated with words, and links represent the possible succession of words. The lattice density is defined as the number of words in the lattice divided by the number of words in the transliteration. JRTk allows for various lattice-related functions, such as filler word removal, as well as beam-width pruning to obtain cleaned lattices in the desired density.

The translation system used is the CMU statistical machine translation (SMT) system, as described above and in more detail in Vogel et al. (2003) and Vogel (2003). The decoder was extended to read entire word lattices. A word lattice is traversed from left to right, and word-to-word and phrase-to-phrase translations are added by extending the lattice with edges to which the translations are attached. The enlarged word lattice represents the search space in which the best path is found, with path scores accumulated over translation model scores, target language model scores, acoustic scores, and source language model scores.

The baseline system uses the single best hypothesis for coupling ASR and MT, and gives a BLEU score of 16.83. The BLEU score is a standard measure of translation quality (Papineni et al., 2002). It is measured as a weighted sum of N-gram precision counts up to $N = 4$, modified by a length penalty. Higher BLEU scores indicate better translation quality.

The first experiment applied lattice-based coupling without taking into account acoustic scores or language model scores of the paths in the JRTk lattice. This was to determine if the SMT system could benefit from those additional paths in the word lattice that have a significantly lower error rate than the first-best hypothesis. However, despite the advantage of the lattice topology with different densities, it did not outperform the baseline. Since the ASR lattice typically contains not only paths that are better than the first-best hypothesis but also many paths that are worse, the decoder can choose a path that is easy to translate—that is, a path that gives high probabilities for the translation model and target language model—but that can have many recognition errors. In other words, the resulting translation is not guaranteed to be a translation of what the speaker originally said. This result indicates that it is necessary to incorporate the ASR acoustic and source language model scores into the selection process of the MT system.

In the second experiment, weighted acoustic scores were added to the translation scores. The BLEU score was calculated for acoustic score weights ranging from 0.01 to 0.29 and for lattices of different densities. The best improvement for the BLEU score over the baseline was 7.3%, obtained with a lattice density of 3 and an acoustic score weight of 0.28. The BLEU score increased from that of the one-best hypothesis with the addition of acoustic scores; however, no smooth transition could be found with increasing densities, which indicates that source language model scores need to be included in the translation system.

A closer analysis showed that the length of an utterance has an impact on the improvements on the BLEU score. We therefore used a development test set to find the best BLEU score improvements given the utterance lengths, acoustic score weights, and lattice densities. Table 10.6 shows the optimal settings for different utterance lengths.

The optimized parameters were finally applied to the test set and resulted in a relative improvement over the baseline of 16.22%.

Finally, both ASR scores, the acoustic and the source language model scores were included in order to identify paths in the word lattice that have few recognition errors and, at the same time, represent good translation hypotheses.

Figure 10.6. demonstrates the effect of adding both ASR scores to the translation model and target language model scores.

For a lattice of density 4 and an ASR score weight of the 0.89, the improvement of the BLEU score over the baseline is 12.71% (on the test set). When adding source language model scores, the system was again tuned separately for sentences of different lengths using a development set.

Table 10.6 Optimal density and acoustic score weight based on utterance length.

	Words	Optimal Acoustic Weight	Optimal Lattice Density
Short	1–5	0.0	8
Medium	6–10	0.22	3
Long	11–23	0.01	2

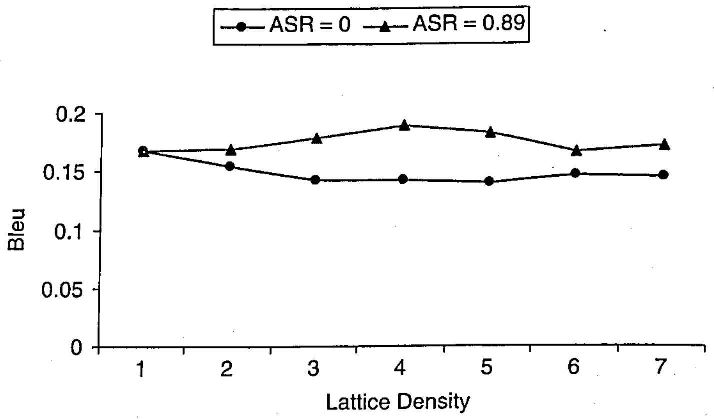


Figure 10.6: Translation quality with and without ASR scores (acoustic model and source language model scores).

The resulting optimal model scaling factors and lattice densities are shown in Table 10.7.

These optimal parameter settings are now much smoother than in the previous case, in which only the acoustic model scores were used, indicating that adding the source language model makes the overall system more robust and stable.

When applying those settings to unseen test data, an improvement of 21.78% in the BLEU score was achieved. Further tuning of the parameters on the test set resulted in a relative improvement of 26.9% over the baseline.

Table 10.8 summarizes the results of the experiments in tight coupling between speech recognition and translation.

Table 10.7 Optimal density and acoustic score weight based on utterance length when using acoustic and source language model scores.

	Words	Optimal Acoustic Weight	Optimal Lattice Density
Short	1–5	0.61	4
Medium	6–10	0.93	2
Long	11–23	0.89	3

Table 10.8 Summary of translation results for tight coupling between recognition and translation (D = lattice density).

	BLEU Score	Improvement
Baseline	16.8	–
Lattice (D = 3)	14.2	−15.5%
with Acoustic Model (D = 3)	18.0	7.3%
- Length specific	19.5	16.2%
with AC and Source LM (D = 4)	18.9	12.7%
- Length specific	20.5	21.8%

10.4 Portable Speech-to-Speech Translation: The ATR System

This section describes example-based and transfer-based approaches to speech-to-speech translation (S2ST) in greater detail and exemplifies their use in a complete speech-to-speech system—namely, the system developed at ATR (Advanced Telecommunications Research Institute International) in Japan. ATR was founded in 1986 as a basic research institute in cooperation with the Japanese government and the private sector, and initiated a research program on Japanese-English speech-2-speech translation (S2ST) soon afterward. This program has addressed not only S2ST approaches proper but also the speech recognition, speech synthesis, and integration components that are required for a complete end-to-end system. The first phase of the program focused on a feasibility study of S2ST, which only allowed a limited vocabulary and clear, read speech. In the second phase, the technology was extended to handle "natural" conversations in a limited domain. The target of the current third phase is to field the S2ST system in real environments. The intended domain of the system is dialog applications.

While earlier phases of the research program were characterized by hybrid rule-based and statistical approaches, the current technology is heavily corpus-based and uses primarily statistical techniques to extract information from linguistically annotated databases. The reason is that corpus-based methods greatly facilitate the development of systems for multiple languages and multiple domains and are capable of incorporating recent innovative technology trends for each component. Domain portability is particularly important, since S2ST systems are often used for applications in a specific situation, such as supporting a tourist's conversation in non-native languages. Therefore, the S2ST technique must include automatic or semiautomatic functions for adapting to specific situations/domains in speech recognition, machine translation, and speech synthesis (Lavie et al., 2001b).

10.4.1 A Corpus-Based MT System

Corpus-based machine translation (MT) technologies were proposed in order to handle the limitations of the rule-based systems that had formerly been the dominant paradigm in machine translation. Experience has shown that corpus-based approaches (1) can be applied to different domains; (2) are easy to adapt to multiple languages because knowledge can be automatically extracted from bilingual corpora using machine learning methods; and (3) can handle ungrammatical sentences, which are common in spoken language. Corpus-based approaches used at ATR include, for example, Transfer-Driven Machine Translation (TDMT) (Furuse and Iida, 1994; Sumita et al., 1999), which is an exampled-based MT system based on the syntactic transfer method. One current research theme is to develop example-based translation technologies that can be applied across a wide range of domains, and to develop stochastic translation technologies that can be applied to language pairs with completely different structures, such as English and Japanese. Example-based methods and stochastic methods each have different advantages and disadvantages and can be combined into a single, more powerful system.

Our overall speech-to-speech translation system is shown in Figure 10.7. The system consists of three major modules: a multilingual

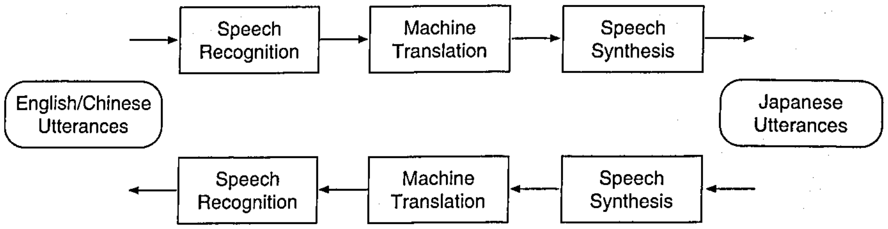


Figure 10.7: Block diagram of the ATR S2ST system.

speech recognition module, a multilingual machine translation module, and a multilingual speech synthesis module. These modules are designed to process Japanese, English, and Chinese using corpus-based methods. Each module is described in more detail below.

Multilingual Speech Recognition

The speech recognition component uses an HMM-based approach with context-dependent acoustic models. In order to efficiently capture contextual and temporal variations in the input while constraining the number of parameters, the system uses the successive state splitting (SSS) algorithm (Takami and Sagayama, 1992) in combination with a minimum description length criterion (Jitsuhiro et al., 2003). This algorithm constructs appropriate context-dependent model topologies by iteratively identifying an HMM state that should be split into two independent states. It then reestimates the parameters of the resulting HMMs based on the standard maximum-likelihood criterion. Two types of splitting are supported: contextual splitting and temporal splitting, as shown in Figure 10.8.

Language modeling in the multilingual speech recognizer is performed by statistical N-gram models with word classes. Word classes are typically established by considering a word's dependencies on its left-hand and right-hand context. Usually, only words having the same left-hand and right-hand context dependence belong to the same word class. However, this

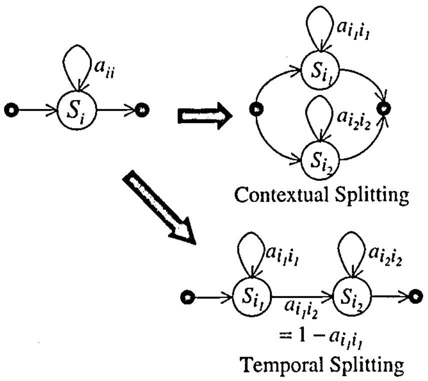


Figure 10.8: Contextual splitting and temporal splitting.

word class definition is not adequate for representing the distribution of words that have the same left-hand or right-hand context but not both, as exemplified by the words "a" and "an." The left-hand context of "a" and "an" is almost equivalent; however, the right-hand context is significantly different. Such differences are more common in languages with inflection, such as French and Japanese. For example, the Japanese inflection form has an influence only on the right-hand context, while the left-hand context can be shared between the same words with different inflection forms.

We therefore use multidimensional word classes to represent left- and right-hand context dependence separately (Yamamoto et al., 2003). Multidimensional word classes can assign the same word class to "a" and "an" to represent the left-context Markovian dependence (left-context class), and assign them to different word classes to represent the right-context Markovian dependence (right-context class). Each multidimensional word class is automatically extracted from the corpus using statistical information, rather than grammatical information such as part of speech. Formally, this is defined as follows:

$$P(w_i|w_{i-N+1}\dots w_{i-1}) = P(C^l(w_i)|C^{rN-1}(w_{i-N+1})\dots C^{r2}(w_{i-2}C^{r1}(w_{i-1})) \\ P(w_i|C^l(w_i)), \tag{10.8}$$

where the suffix for class C is used to represent position-dependent (left- and right-context) Markovian dependence. Here, $C^l(w)$ represents the left-context class to which the word w belongs, and $C^{ri}(w)$ represents the right-context class to which the ith word w belongs. Hereafter, we refer to these class N-grams based on multidimensional classes as multiclass N-grams.

Multilingual Machine Translation

The translation engine (named C-cube, for "corpus-centered computation") relies heavily on corpus-based technology. Translation knowledge is extracted from corpora, translation quality is gauged and optimized by reference to corpora, and the corpora themselves are paraphrased or filtered by automated processes. Figure 10.9 shows an overview of the machine-translation system developed in the C-cube project.

There are two main approaches to corpus-based machine translation: (1) Example-Based Machine Translation (EBMT) (Nagao, 1984; Somers,

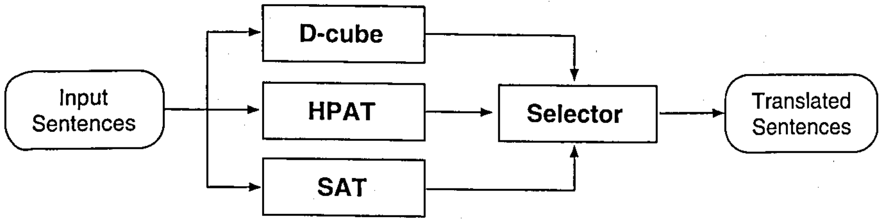


Figure 10.9: An overview of the machine translation system developed in the C-cube project.

1999); and (2) Statistical Machine Translation (SMT) (Brown et al., 1993; Knight, 1997; Ney, 2001; Alshawi et al., 2000; Wang and Waibel, 1998; Och et al., 1999; Venugopal et al., 2003), which was already described above. C-cube develops both technologies in parallel and blends them into a single system. Three different machine-translation engines have been developed in the course of this project: D-cube, HPAT, and SAT. D-cube is a sentence-based EBMT engine. It retrieves the most similar example of the input from example sentences using dynamic programming based matching, and adjusts the gap between the input and the retrieved example by using dictionary information (Sumita, 2001). HPAT is a phrase-based EBMT engine.. Based on phrase-aligned bilingual trees, transfer patterns are generated. According to these patterns, the source phrase structure is obtained and converted to generate target sentences (Imamura, 2002). SAT is the Statistical ATR Translator. It translates between Japanese and English and builds on a word-based SMT framework. SAT is a developing series of SMT models, which includes phrase-based translation (T. Watanae and Sumita, 2002), chunk-based translation (Y. Akiba and Sumita, 2002), and sentence-based greedy decoding (Watanae and Sumita, 2003). These three systems have obtained good performance on Japanese and English translation, and have recently been successfully applied to Japanese and Chinese.

Sentence-Based EBMT

The ATR sentence-based EBMT approach relies on D3—a dynamic programming transducer that exploits matching between word sequences

(Sumita, 2001). Suppose we are translating a Japanese sentence into English. The Japanese input sentence (1-j) is translated into the English sentence (1-e) by utilizing the English sentence (2-e), whose source sentence (2-j) is similar to (1-j). The common parts are unchanged, and the different portions, shown in boldface, are substituted by consulting a bilingual dictionary.

;;; A Japanese input
(1-j) **iro**/ga/ki/ni/iri/masen
;;; the most similar example in corpus
(2-j) **dezain**/ga/ki/ni/iri/masen
(2-e) I do not like the **design**.
;;; the English output
(1-e) I do not like the **color**.

We retrieve the most similar source sentence example from a bilingual corpus. For this, we use *DP-matching*, which computes the *edit distance* between word sequences while simultaneously identifying the matched portions between the input and the example. The edit distance is calculated by summing the count of the inserted words, the count of the deleted words, and the semantic distance value of the substituted words, normalized by the sum of the lengths of the input and the source part of the translation example. The semantic distance between two substituted words is calculated by using the hierarchy of a thesaurus (Sumita and Iida, 1991a). Thus, the language resources required in addition to a bilingual corpus are a bilingual dictionary, which is used for generating target sentences, and thesauri of both languages, which are used for incorporating the semantic distance between words into the distance between word sequences. Furthermore, lexical resources are used for word alignment.

Phrase-Based EBMT

The second EBMT system is different from the first in that it parses bitexts of a parallel corpus with grammars for both source and target languages. It incorporates a new phrase alignment approach called Hierarchical Phrase Alignment (HPA), which was proposed by Imamura (2001). HPA retrieves equivalent phrases that satisfy two conditions: (1) words in the pair correspond with no deficiency and no excess; and (2) the phrases are of the same syntactic category. This was subsequently extended to HPA-based translation (HPAT) (Imamura, 2001). HPAed bilingual trees include all information necessary to automatically generate transfer patterns. Translation is done according to transfer patterns using a Transfer-Driven MT (TDMT) engine (Furuse and Iida, 1996). Finally, a *feedback cleaning* method (Imamura, 2003) is applied that utilizes automatic evaluation to remove incorrect/redundant translation rules. The standard BLEU method was utilized to measure translation quality for the feedback process, and a hill-climbing algorithm was applied to search for the combinatorial optimization. Finally, incorrect/redundant rules were removed from the set of all rules initially acquired from the training corpus, which improved the translation quality considerably.

Phrase alignment refers to the extraction of equivalent partial word sequences between bilingual sentences. The term "phrase alignment" is used since these word sequences include not only words but also noun phrases, verb phrases, relative clauses, and so on.

For example, when the following bilingual sentence is given,

English: *I have just arrived in New York.*
Japanese: *Nyuyooku ni tsui ta bakari desu.*

the phrase alignment should extract the following word sequence pairs.

- *in New York* ↔ *Nyuyooku ni*
- *arrived in New York* ↔ *Nyuyooku ni tsui*
- *have just arrived in New York* ↔ *Nyuyooku ni tsui ta bakari desu*

We call these *equivalent phrases* and define this task as extracting phrases that satisfy the following two conditions.

Condition 1 (Semantic constraint):
Words in the phrase pair correspond to no deficiency and no excess.
Condition 2 (Syntactic constraint):
The phrases are of the same syntactic category.

In order to extract phrases that satisfy the two conditions, corresponding words—called *word links*, represented as $WL(word_e, word_j)$—are first extracted by word alignment. Next, the sentence pair is parsed and phrases and their syntactic categories are acquired. Those phrases—which include some word links, exclude other links, and are of the same syntactic categories—are regarded as equivalent.

For example, in the case of Figure 10.10(a), `NP(1)` and `VMP(2)` are regarded as equivalent because they only include *WL(New York, Nyuyooku)* and are of the same syntactic category. In the case of *WL(arrived, tsui)*, `VP(3)` is regarded as equivalent, and in the case of both word links, `VP(4)`, `AUXVP(5)`, and `S(6)` are regarded as equivalent. Consequently, six equivalent phrases are extracted hierarchically.

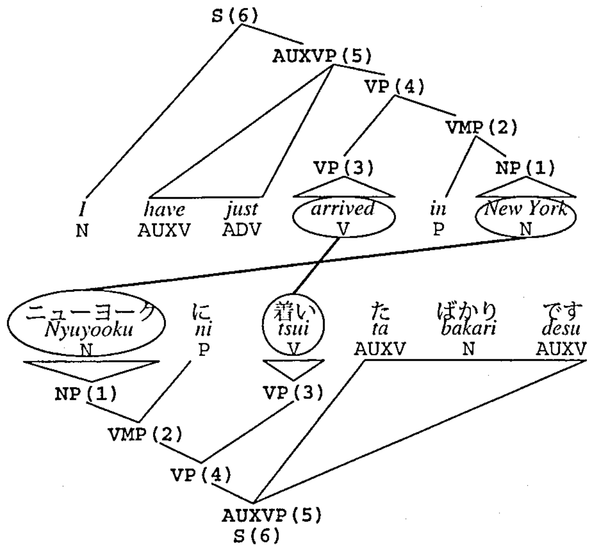
Even though word links are available, the parts-of-speech (POS) of the words are sometimes different in different languages, as shown in the second example in Figure 10.10(b). In this case, the phrases that contain only *WL(fully, ippai)* or only *WL(booked, yoyaku)* are not regarded as equivalent because of the syntactic constraint, and `VP(2)` nodes are extracted first. Thus, few unnatural short phrases are extracted as equivalent.

The problem besetting the method described above is that the result of the phrase alignment directly depends on the parsing result. This problem can be solved by using the following features and techniques.

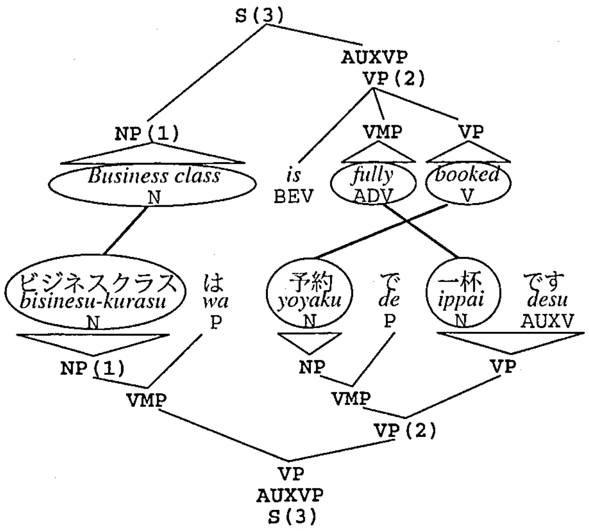
Disambiguation Using Structural Similarity: As Kaji et al. (1992) and Matsumoto et al. (1993) showed, some parsing ambiguities can be resolved when the two languages are made to correspond. This disambiguation utilizes structural similarity. For example, a PP attachment in English is ambiguous as to whether it modifies a noun or a verb, but this is nearly always definite in Japanese. Hence, when the attachment is assumed to modify the same word, the ambiguity is resolved. Accordingly, the structures between the two languages become similar.

We employ a *phrase correspondence score* to measure structural similarity. This measure is calculated by counting the phrases that satisfy the above two conditions. The parsing candidate that has the maximal score is selected.

Combination of Partial Trees: Partial parsing is an effective way to avoid a lack of grammar or to parse ungrammatical sentences. It is used to combine partial candidates in the parser. Therefore, a criterion as to whether the part is valid or not is necessary for the combining process. We utilize the phrase correspondence score as the criterion, and a partial tree sequence that maximizes the sum of the phrase correspondence scores is searched for.



(a) Example of Simple Translation



(b) Example of Different POS Translation

Figure 10.10: Examples of Hierarchical Phrase Alignment (top and bottom trees denote English and Japanese, respectively; lines between languages denote word links).

Syn. Cat.	Source Pattern	Target Pattern	Source Example
VP	X_{VP} *at* Y_{NP} $\Rightarrow$	Y' *de* X'	*(present, conference)*
		Y' *ni* X'	*(stay, hotel), (arrive, p.m.)*
		Y' *wo* X'	*(look, it)*
NP	X_{VP} *at* Y_{NP} $\Rightarrow$	Y' *no* X'	*(man, front desk)*

Figure 10.11: Examples of transfer rules in which the constituent boundary is "at."

The forward DP backward A^* search algorithm (Nagata, 1994) is employed to speed up the combination.

Transfer Driven Machine Translation (TDMT)

The Transfer Driven Machine Translation system, or TDMT (Furuse and Iida, 1994; Sumita et al., 1999), used here is an example-based MT system (Nagao, 1984) based on the syntactic transfer method (called transfer-based MT).

Transfer rules of TDMT represent the correspondence between source and target language expressions. These are the most important kinds of knowledge in TDMT. Examples are shown in Figure 10.11 that include the preposition "at." In this rule, source language information is constructed by a source pattern and its syntactic category. The source pattern is a sequence of variables and constituent boundaries (functional words or part-of-speech bigram markers). Each variable is restricted by a syntactic category using daughter rules. Namely, source language information is equivalent to a context-free-grammar such that the right side of each rewrite rule absolutely contains at least one terminal symbol.

Target patterns are similarly constructed with variables and constituent boundaries, but they do not have POS bigram markers. In addition, each rule has source examples, which are instances of variables. The source examples are head-words acquired from training sentences. For instance, the first rule of Figure 10.11 means that the English phrase "present at (the) conference" was translated into the Japanese phrase "*kaigi* (conference) *de happyo-suru* (present)."

At the time of translation, the source sentence is parsed using source patterns. Then, the target structure, which is mapped by target patterns, is generated (Figure 10.12). However, as shown in Figure 10.11, one transfer

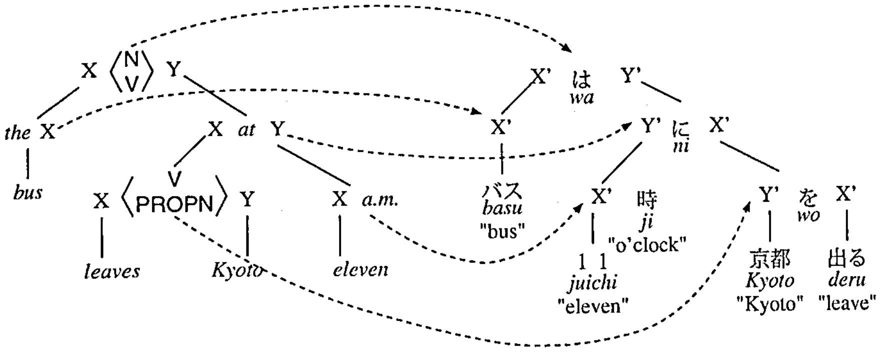


Figure 10.12: Example of TDMT transfer process.

rule has multiple target patterns. In order to select the appropriate target pattern, semantic distances (node distances on the thesaurus; refer to Sumita and Iida (1991b)) are calculated between the source examples and the daughter head-words of the input sentence, and the target pattern corresponding to the best matching example is selected. Therefore, each rule also has head information.

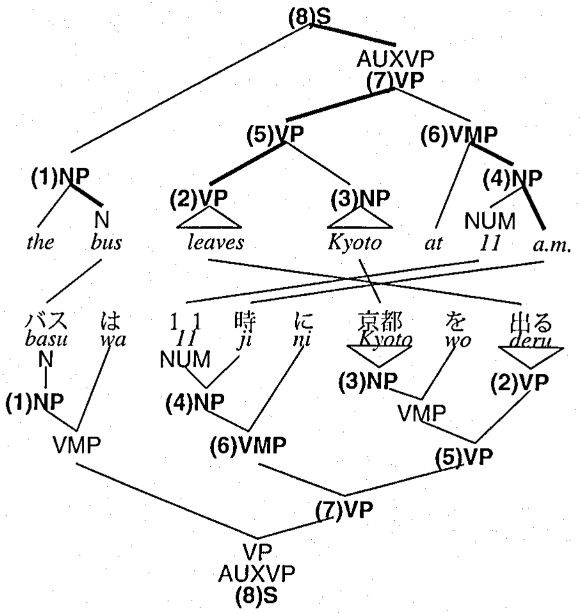
For example, when the input sentence "The bus leaves Kyoto at eleven a.m." is given, the source pattern (X *at* Y) is used. Then, the head-word of the variable X is "*leave*" and Y is "*a.m.*" According to the semantic distance calculation, the source example (*arrive*, *p.m.*) is the closest match. Therefore, the target pattern (Y' *ni* X') is selected. The semantic distance is also applied to parsing disambiguation.

Application of HPA Results for TDMT

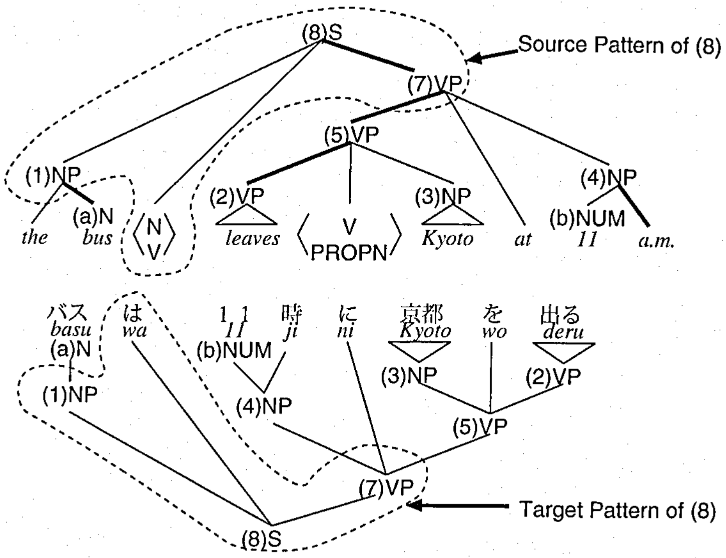
The transfer rules described previously are constructed by source patterns that include their syntactic category, target patterns, source examples, head information, and local dictionaries. They are generated as follows from the HPA results (Figure 10.13).

1. First, the result of HPA is transformed into a structure that can construct transfer rules.

    - If an input word sequence includes continuous content words, insert a bigram marker in the intermediate of the content words.



(1) Result of HPA (Bold lines denote head information)



(2) Tree structure after transformation

Figure 10.13: Example of transfer rule generation.

The bigram marker is an artificial word, which works as a functional word when the translator parses the input sentence.

- If the edges of a word link are content words and are of the same POS types, a new word-level correspondence is added. The function of this correspondence is to translate unseen words by referring to the translation dictionary.

  In Figure 10.13, the correspondences `(a)N` and `(b)NUM` are supplied from the word links *WL(bus/*N, *basu/*N) and *WL(11/*NUM, *11/*NUM).
- When the source pattern is generated, the correspondence is removed if variables are continuous because TDMT does not accept the series of variables. In Figure 10.13, `(6)VMP` is removed.
- All nodes that do not have correspondences are removed except for the top node.

2. Next, source patterns, target patterns, source examples, head information, and local dictionaries are created as follows.

   - Source patterns and target patterns are generated from the correspondences. The patterns are generalized by regarding daughter corresponding nodes as variables.
   - Head information is acquired from grammar, and source examples are identified by tracing the parsing tree to the head branch.
   - Local dictionaries are created by word links and by extracting leaf equivalent phrases in which the source phrase contains only a word.

In addition, because the inputs of phrase alignment are aligned sentences, sentence correspondences are added to the phrase alignment results as equivalent when the top nodes of the trees do not have a correspondence.

When the result of HPA is given (as shown in Figure 10.13), five rules are generated (as shown in Figure 10.14). Note that rules are not generated from the correspondences `(2)VP`, `(3)NP`, `(a)N`, and `(b)NUM` because they are output to the local dictionaries.

Syn. Cat.	Source Pattern		Target Pattern	Source Example
(8) S	X_{NP}<N-V>Y_{VP}	⇒	X' *wa* Y'	(*bus, leave*)
(7) VP	X_{VP} *at* Y_{NP}	⇒	Y' *ni* X'	(*leave, a.m.*)
(5) VP	X_{VP}<V-PROPN>Y_{NP}	⇒	Y' *wo* X'	(*leave, Kyoto*)
(1) NP	*the* X_N	⇒	X'	(*bus*)
(4) NP	X_{NUM} *a.m.*	⇒	X'*ji*	(*11*)

Figure 10.14: Example of generated rules from the sentence "The bus leaves Kyoto at 11 a.m."

SAT (Statistical ATR Translator)

As described above the framework of statistical machine translation formulates the problem of translating a sentence of language J into another language E as the maximization problem of the conditional probability $\hat{E} = \text{argmax}_E P(E|J)$ (Brown et al., 1993b). The application of the Bayes rule resulted in $\hat{E} = \text{argmax}_E P(E)P(J|E)$. The former term $P(E)$ is called a language model, representing the likelihood of E. The latter term $P(J|E)$ is called a translation model, representing the generation probability from E into J.

Under this concept, Brown et al. (1993b) presented a translation model in which a source sentence is mapped to a target sentence with the notion of word alignment.[1] Although it has been successfully applied to similar language pairs, such as French-English and German-English, little success has been achieved for drastically different language pairs, such as Japanese-English. The problem lies in the huge search space resulting from frequently occurring insertion/deletion, the larger numbers of fertility for each word, and the complicated word alignments. Due to its complexity, a beam search decoding algorithm often leads to suboptimal solutions.

Word alignment based statistical translation represents bilingual correspondence by the notion of word alignment A, allowing one-to-many generations from each source word. A is an array for target words describing the indices to the source words. For instance, Figure 10.15 illustrates an example of English and Japanese sentences, E and J, with sample word alignments A. In this example, the "show$_1$" has generated two words,

[1]The source/target sentences are the channel model's source/target that correspond to the translation system's output/input.

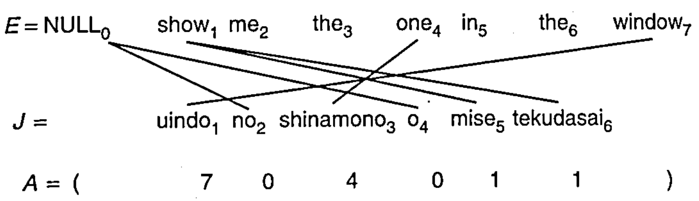

E = NULL$_0$ show$_1$ me$_2$ the$_3$ one$_4$ in$_5$ the$_6$ window$_7$
J = uindo$_1$ no$_2$ shinamono$_3$ o$_4$ mise$_5$ tekudasai$_6$
A = (7 0 4 0 1 1)

Figure 10.15: Example of word alignment.

"mise$_5$" and "tekudasai$_6$." The word alignment assumption, the translation model $P(J|E)$, can be further decomposed without approximation:

$$P(J|E) = \sum_A P(J,A|E)$$

The word alignment based statistical translation model was originally intended for similar language pairs, such as French and English. When applied to Japanese and English, for instance, the resulting word alignments are very complicated, as seen in Figure 10.15. The complexity is directly reflected by the structural differences—that is, English takes an SVO structure while Japanese usually takes the form of SOV. In addition, insertion and deletion occur very frequently as seen in the example.

Example-Based Decoder

Instead of decoding word-by-word and generating an output string word-by-word, as seen in beam search strategies, Watanabe and Sumita (2003) proposed an alternative strategy taken after the framework of example-based machine translation: Retrieve a translation example from a parallel corpus whose source part is similar to the input sentence, then slightly modify the target part of the example so that the resulting part becomes the actual translation (see to Figure 10.16).

Retrieval of Translation Examples

Given an input sentence J_0, the retrieval process looks up a collection of translation examples $\{(J_1, E_1), (J_2, E_2), \dots\}$, where J_k is similar to J_0 using

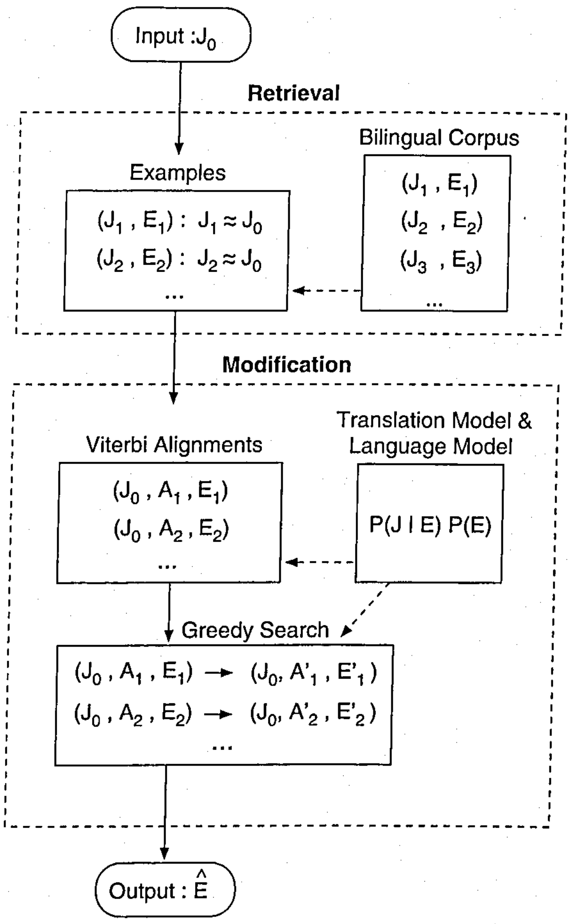


Figure 10.16: Example-based decoding.

the edit distance criteria, penalizing an insertion/deletion/substitution with one editing step. A simple implementation of the multiple alignment problem resulted in an NP-hard problem in which Dynamic Programming (DP) algorithms should be applied to all the examples in a bilingual corpus. An additional problem results from the fact that the DP matching criteria does not reflect the closeness of two sentences. For instance, the sentence "I'm a computer system engineer" can match many examples—such as "I'm a graduate student" or "I'm an engineer"—with the same edit distance of 3.

In order to overcome those problems, a tf/idf criterion was introduced to search for the relevant examples by treating each example as a document. Particularly, when given an input J_0, the decoder first retrieves $N_r(\leq N)$ relevant translation samples $\{(J_1, E_1), (J_2, E_2), \ldots\}$ using the tf/idf criterion, as seen in the information retrieval framework (Manning and Schütze, 1999):

$$P_{tf/idf}(J_k, J_0) = \sum_{i:J_{0,i} \in J_k} \frac{\log(N/df(J_{0,i}))/\log N}{|J_0|},$$

where $J_{0,i}$ is the ith word of J_0, $df(J_{0,i})$ is the document frequency for the word $J_{0,i}$, and N is the total number of examples in a bilingual corpus. Note that the frequency is set to 1 if the word exists in J_k, otherwise 0; and tf/idf scores are summed and normalized by the input sentence length.

Then, for each sample (J_k, E_k), DP matching is performed against J_0 to compute the edit distance:

$$dis(J_k, J_0) = I(J_k, J_0) + D(J_k, J_0) + S(J_k, J_0),$$

where $k \leq N_r$ and $I(J_k, J_0)$, $D(J_k, J_0)$, and $S(J_k, J_0)$ are the number of insertions, deletions, and substitutions, respectively. All samples are scored by the following criteria:

$$score = \begin{cases} (1.0 - \alpha)(1.0 - \frac{dis(J_k, J_0)}{|J_0|}) \\ \quad +\alpha P_{tf/idf}(J_k, J_0) & dis(J_k, J_0) > 0 \\ 1.0 & \text{otherwise} \end{cases}$$

In this scoring, $dis(J_k, J_0)$ is transformed into the word error rate by normalization with the input length $|J_0|$, then subtracted from 1 to derive the correction rate. The correction rate is linearly interpolated with the normalized tf/idf score with a tuning parameter α that was set to 0.2 in the experiments. Note that when the distance of the input sentence and the source part of an example is zero, the example is treated as an exact match and is scored as one.

Modification of Translation Examples

After the retrieval of similar examples $\{(J_1, E_1), (J_2, E_2), \ldots\}$, the modification step tunes the sample translations according to a statistical translation model. In this step, the greedy algorithm was applied, originated in Germann et al. (2001). However, it differs in that the search starts from the retrieved translation example, not from a guessed translation.

For each translation example (J_k, E_k),

1. Compute the Viterbi alignment A_k for the pair (J_0, E_k)
2. Perform the hill-climbing algorithm for (J_0, A_k, E_k) to obtain (J_0, A'_k, E'_k) by modifying A_k and E_k

A_k is computed through hill climbing by rearranging particular word alignments as proposed by Brown et al. (1993b). When the retrieved samples contain an exact match scored as one, the search terminates and returns the retrieved examples with the highest probability together with the Viterbi alignment.

When the samples are not an exact match, the decoder performs hill climbing, modifying the output and alignment for a given example (J_0, A, E), where A is the word alignment initially assigned by the Viterbi alignment and E is the target part of the example. In this greedy strategy, the operators applied to each hill-climbing step are:

- Translate words: Modify the output word E_{A_j} to e aligned from J_{0j}. If $e = \text{NULL}$ then J_{0j} is aligned to NULL and $A_j = 0$. When the fertility of E_{A_j} becomes zero, the word E_{A_j} is removed. e is selected from among the translation candidates, computed from the inverse of the lexicon model (Germann et al., 2001).
- Translate and insert words: Perform the translation of a word, and insert a sequence of zero fertility words at appropriate positions. The candidate sequence of zero fertility words is selected from the Viterbi alignment of the training corpus (Watanabe and Sumita, 2002).
- Translate and align words: Move the alignment of A_j to i and modify the output word from E_i to e.
- Move alignments: This operator does not alter the output word sequence, but modifies the alignment A through moving/swapping (Brown et al., 1993b).

- Swap segments: Swap nonoverlapping subsets of E by swapping a segment from i_0 to i_1 and from i_2 to i_3. Note that $i_1 < i_2$.
- Remove words: Remove a sequence of zero fertility words from E.
- Join words: Join words of E_i and $E_{i'}$ when the fertility of both of the words is more than or equal to one.

For each hill-climbing step, the decoder tries all the possible operators, then selects the best step for the next iteration. The hill-climbing operators were taken from Germann et al. (2001), but two new operators were added: the "translate-and-align-words," and the "move-alignment." If at the first step of computing the Viterbi alignment an input word is found for which no translation exists among the retrieved samples, this word will either be aligned to NULL or to an irrelevant word, raising the fertility. Therefore, the translate-and-align operator can force it to move the alignment to another word and to choose the right word-for-word translation using the lexicon model. Similarly, the move-alignment operator can resolve the problem by simply alternating the existing word alignments.

Selector Approach

Today, no single system can translate everything. The translation quality changes from sentence to sentence and system to system. Since each system has its own strengths for translating particular kinds of sentences, the differences in quality can be substantial. The translations provided by multiple engines are often complementary. Thus, we could obtain a large increase in accuracy if it were possible to select the best out of different translations for each input sentence.

We adopted a selector approach, as shown in Figure 10.9, by using both the language and translation models of SMT (Akiba et al., 2002). The selector outperformed conventional selectors using only the language model (the target N-gram in the experiment with the three previously mentioned machine translation systems).

Although it is likely that the selector based on SMT models always favors SMT, that was not true for this experiment. This suggests that although an SMT engine fails to produce good translation under time and space restrictions, SMT models can be used as a good measure for comparing the quality of multiple translations. The second suggestion is amplified to a general framework in that a translation system can be divided into two parts: (1) a generator of translation candidates and (2) a selector of the best one by using automatic quality evaluation. This contrasts with a conventional system, which often determines a single translation in a deterministic fashion. The emphasis here lies on focus on the independence of quality evaluation as an automated process, and the fact that an SMT model works well as an evaluator.

The selector scheme is an easy-to-implement method of improving overall performance, since there is no need to investigate the complex relationships between the resources and processes of the component MT systems by hand. This simplifies the development process because each element MT system can be improved without any consideration of the other components, and this loosely coupled system automatically boosts the performance of the whole system by exploiting the elemental improvements.

Multilingual Text-to-Speech

The synthesis component of the S2ST system is based on the concatenative synthesis approach (see Chapter 7). As one of the pioneers in corpus-based speech synthesis technology, ATR has made contributions to the progress of the technology through various studies, which have led to the development of three text-to-speech (TTS) systems, namely ν-talk (Sagisaka et al., 1992b), CHATR (Black and Taylor, 1994), and XIMERA, which is currently under development. The framework of XIMERA is essentially the same as that of CHATR, ν-talk, and other corpus-based TTS systems. The general aim is to achieve a substantial improvement in naturalness through the optimization of the component technologies of TTS.

A block diagram of XIMERA is shown in Figure 10.17. Similar to most concatenative TTS systems, XIMERA is composed of four major modules, namely a text processing module, a prosodic parameter generation module, a segment selection module, and a waveform generation module.

The target languages of XIMERA are Japanese and Chinese. Although the framework of corpus-based synthesis is language independent, most modules, in reality, must be developed or tuned toward the target language. Language dependent modules include the text processing module, acoustic models for prosodic parameter generation, speech corpora, and the cost

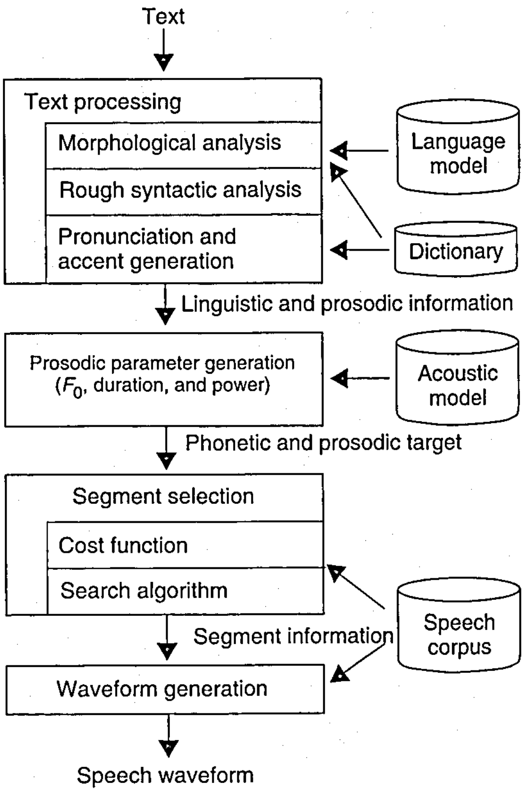


Figure 10.17: A block diagram of a TTS module.

function for segment selection. The search algorithm for segment selection is also related to the target language via the cost function.

Although emotion and speaking style variations are indispensable for achieving a speech synthesizer that can be used in place of humans, XIMERA is currently focused on a normal reading speech style suitable for news reading and emotionless dialog between man and machine.

The prominent features of XIMERA are: (1) its large corpora (a 110-hour corpus of a Japanese male, a 60-hour corpus of a Japanese female, and a 20-hour corpus of a Chinese female); (2) HMM-based generation of prosodic parameters (Tokuda et al., 2000); and (3) a cost function for segment selection, optimized on perceptual experiments (Toda et al., 2004).

Text Processing and Speech Corpora

The text processing module consists of three submodules for morphological analysis, rough syntactic analysis, and pronunciation and accent generation. The morphological analysis is conducted based on a bigram language model and a morpheme dictionary consisting of 239,591 Japanese and 195,959 Chinese entries. The rough syntactic analysis determines (1) a dependency between adjacent words, which is mainly used for F_0 generation, and (2) clause boundaries, which are mainly used for pause insertion. The pronunciation generation determines the readings of homographs and euphonic changes of unvoiced to voiced sounds. The accent generation determines the accent type of an accentual phrase based on accent types and the accent concatenation features of the constituent morphemes. In terms of the dichotomy of corpus-based versus noncorpus based, the first two subprocesses are corpus-based technologies while the latter one is not.

Three large-scale single-speaker speech corpora were collected. One corpus of Japanese male of 111-hour length, one corpus of Japanese female of 60-hour length, and one corpus of Chinese female of 20-hour length. The contents of the Japanese corpora include news, novels, and travel conversations. The travel conversations were uttered in a reading style. The given corpus size in hours include speech pauses within utterances but not between utterances. The speakers were professional narrators. The recordings took 181 days over a span of 973 days (Japanese male corpus), 95 days over a span of 307 days (Japanese female corpus), and 32 days over a span of 63 days (Chinese female corpus). The speech was recorded in a soundproof room. To obtain high signal-to-noise ratio (SNR), a large diaphragm condenser microphone with a cardioid directional pattern was used. The speech data were digitized at a sampling frequency of 48 kHz with 24-bit precision, recorded onto a hard disk. After reading errors were removed by human inspection, speech data were separated into utterances, high-pass filtered at 70 Hz, precision-converted down to 16 bits after amplitude adjustment, and stored in files. Phonemic transcriptions in katakana characters were also inspected and corrected by humans. Phone segmentation was conducted automatically by using speaker-dependent monophone HMMs. The results were not corrected manually.

Generation of Prosodic Parameters

Prosodic parameters, namely F_0, phone duration, and power, are generated applying the HMM-based speech synthesis algorithm (Tokuda et al., 1995, 2000). Japanese speech is modeled with context-dependent HMMs of 42 phonemes consisting of five states each, while Chinese speech is modeled with context-dependent HMMs of 60 initials and finals. The generated prosodic parameters are sent to the succeeding module to be used as targets for segment selection.

Cost Function

The cost function of a sentence for segment selection is given by

$$C_g = \frac{1}{N} \sum_{i=1}^{N} C_l(u_i, t_i)^p, \tag{10.9}$$

where N denotes the number of targets in the sentence, C_l denotes a local cost at the target t_i, and u_i and t_i, respectively, denote the ith target and segment candidate. Power p was determined to be 2 as a result of perceptual experiments (Toda et al., 2003). The local cost is given by

$$\begin{aligned} C_l(u_i, t_i) = {} & w_{F0} \cdot C_{F0}(u_i, t_i) + w_{dur} \cdot C_{dur}(u_i, t_i) \\ & + w_{cen} \cdot C_{cen}(u_i, t_i) + w_{F0c} \cdot C_{F0c}(u_i, t_i) \\ & + w_{env} \cdot C_{env}(u_i, t_i) + w_{spg} \cdot C_{spg}(u_i, t_i), \end{aligned} \tag{10.10}$$

where $C_{F0}(u_i, t_i)$, $C_{dur}(u_i, t_i)$, and $C_{cen}(u_i, t_i)$, respectively, denote errors in F_0, segment duration, and spectral centroid between the target and a segment candidate (target costs). On the other hand, $C_{F0c}(u_i, t_i)$, $C_{env}(u_i, t_i)$, and $C_{spg}(u_i, t_i)$, respectively, denote discontinuities of F_0, phonetic environment, and spectrum between adjacent segments (concatenation costs). w_{F0}, w_{dur}, w_{cen}, w_{F0c}, w_{env}, and w_{spg} are corresponding weights for the local costs. Mappings from acoustic measures into the above local costs and weights were optimized based on perceptual experiments (Toda et al., 2004).

10.4.2 Multilingual Corpora

ATR has been constructing three different types of corpora in the travel domain in addition to speech databases for training acoustic models: (1) a large-scale multilingual collection of basic sentences that covers many topics in travel conversations, called BTEC (Basic Travel Expression Corpus) (Takezawa et al., 2002); (2) a small-scale bilingual collection of spoken sentences that reflects the characteristics of the spoken dialogs, called SLDB (Spoken Language Database) (Morimoto et al., 1994); and (3) a small-scale corpus of the MT-assisted dialog, called MAD (Kikui et al., 2003). The first one is used to train the multilingual translation component; the second one is used to link spoken sentences to basic sentences, and the third one is used mainly for evaluation of S2ST.

Basic Travel Expression Corpus (BTEC)

The Basic Travel Expression Corpus (BTEC) was designed to cover utterances for all potential topics in travel conversations, together with their translations (Kikui et al., 2003). Since it is almost infeasible to collect them through transcribing actual conversations or simulated dialogs, sentences from the memories of bilingual travel experts are used. We started by investigating phrase books that contain bilingual (in our case Japanese-English) sentence pairs that experts consider useful for tourists traveling abroad. We collected these sentence pairs and rewrote them to make translations as context independent as possible and to comply with our speech transcription style. Sentences outside of the travel domain or those containing very special meanings were removed.

Table 10.9 shows the basic statistics of the BTEC collections, called BTEC1, 2, and 3. Each collection was created with the same procedure in a different time period. We used a morpheme as the atomic linguistic unit for Japanese (instead of a word), since the morpheme unit is more stable than the word unit. This table shows that the BTEC sentences are relatively short and contain duplications.

Since the BTEC sentences did not come from actual speech conversation, we were able to efficiently create a broad coverage corpus; however, it has two potential problems. First, the corpus may lack utterances that appear in real conversation. For example, when people ask the way to a

Table 10.9 Size of BTEC and SLDB.

	BTEC1	BTEC2	BTEC3	Total (BTEC)	SLDB
Utterances (E)					
# of tokens	175,512	46,288	198,290	420,070	16,110
# of types	102,869	37,791	141,504	254,766	14,266
Words (E)					
# of tokens	1,089,570	311,537	1,316,188	2,717,295	199,951
# of types	14,725	11,796	22,925	27,998	4,544
Words per Utterance (E)	6.21	6.73	6.64	6.47	12.75
Utterances (J)					
# of tokens	172,468	46,268	198,290	417,026	16,110
# of types	108,612	36,869	143,454	264,401	14,259
Morphemes (J)					
# of tokens	1,181,763	341,353	1,434,175	2,957,291	204,894
# of types	20,363	15,081	31,155	39,316	5,765
Morphemes per Utterance (J)	6.85	7.38	7.25	7.09	12.75

bus stop, they often use a sentence like (1). However, BTEC1 contains (2) instead of (1).

(1) I'd like to go downtown. Where can I catch a bus?
(2) Where is a bus stop (to go downtown)?

The second problem is that the frequency distribution of the corpus may be different from the "actual" one. In this corpus, the frequency of an utterance (indirectly) corresponds to how many travel experts came up with this sentence and in how many situations they think the sentence will appear. Therefore, it is possible to think of this frequency distribution as a first approximation of reality, but this is an open question.

BTEC was used as test sentences for evaluating various S2ST systems at IWSLT (International Workshop for Spoken Language Translation), held at ATR in October 2004.

ATR is expanding the volume of BTEC by about one million English-Japanese sentence pairs and creating Chinese-Japanese sentence pairs by translating from English/Japanese to Chinese.

Spoken Language Database (SLDB)

For a comparison of the features of expressions in BTEC and utterances in real conversation, we employed an existing corpus, SLDB. SLDB consists of simulated dialogs between a hotel clerk and a tourist. We used a Japanese speaker and an English speaker to achieve some dialog tasks (e.g., to book a hotel room) through conversation mediated by two professional interpreters: one translating from Japanese to English, the other from English to Japanese. Thus, this corpus can be seen as a collection of bilingual dialogs using ideal S2ST systems.

The statistics of SLDB corpora are also seen in Table 10.9. The number of utterance tokens are the same for each language since the utterances were translated simultaneously.

The travel conversation task was selected for the SLDB corpus because of its familiarity to people and its expected use in future speech translation systems. The interpreters speak English and Japanese in all of the conversations and take the role of the speech translation system. One remarkable characteristic of the database is the integration of speech and linguistic data. Each conversation comprises recorded speech, transcribed utterances, and their correspondences.

In creating SLDB, we have tried to collect conversations consisting of speech and sentences that are acceptable for a speech translation system in the near future. For that reason, we have imposed the following constraints on utterances and turns.

1. A speaker can only speak when it is his/her turn. When a speaker wants to stop speaking and listen to a response, he/she actively transfers the speaking permission to the other speaker.
2. Each utterance must be concluded in ten seconds or less, and is then sent to an interpreter.
3. The interpreter translates the speaker's utterance and conveys it to the other speaker. The speaker's utterance is translated consecutively rather than simultaneously, so that the length of the interpreter's utterance is almost the same as the speaker's original utterance. This is also done to resemble the output of speech translation systems in the near future.
4. When a speaker cannot finish what he/she wants to say within the ten seconds, he/she must use several utterances of ten seconds or less.

Each utterance (or translation) continues until the speaker transfers his/her speaking permission (or alike).

5. No interruption of a speaker (or interpreter) is allowed.

With the above constraints, we can avoid extremely long sentences and overlapping of utterances. All human interpreters were professionals resulting in a high translation quality that never caused mistranslations during the conversations.

According to our transcribed bilingual text, all of the expressions in both English and Japanese are acceptable if we consider that they are spoken languages.

As previously noted, SLDB can be seen as a collection of bilingual dialogs using ideal S2ST systems. However, utterances in SLDB have different characteristics from utterances collected from conversation in a real environment. This can be observed when comparing SLDB to a Japanese monolingual travel conversation database, "The Travel Arrangement Task (TRA)," with conversations collected using similar scenario settings and the same recording conditions for robust speech recognition research (Nakamura et al., 1996; Takezawa et al., 1998a). Table 10.10 shows the characteristics of the bilingual and monolingual travel conversation databases. The frequency of filled pauses and the frequency of self-repairs of the bilingual travel conversation database are less than those of the monolingual travel conversation database. Several constraints may

Table 10.10 Characteristics of bilingual and monolingual travel conversation databases.

Conversation style	Bilingual (J–E)	Monolingual (J–J)
Number of collected conversations	618	892
Speaker participants	71	499
Interpreter participants	23	0
Total number of utterances	16,107	22,874
Total number of Japanese words	301,961	491,159
Utterances including one filled pause or more	24%	42%
Utterances including one self-repair or more	3%	6%
Average unit length (per utterance)	30 morae	35 morae

cause these significant differences. The situations in the Japanese monolingual conversation database involve direct communications between two Japanese speakers, but the bilingual situations involve indirect communications. When collecting conversations, the turn-around time is much longer for the bilingual case than for the monolingual case, such that the speakers in the bilingual case may have more than enough time to consider what to say and how to say their next utterances.

MT-Assisted Dialogs (MAD)

The last approach is intended to collect representative utterances that people will input to S2ST systems. For this purpose, we carried out simulated (i.e., role-play) dialogs between two native speakers of different mother tongues using a Japanese-English bidirectional S2ST system instead of human interpreters. We replaced the speech recognition modules with human typists for most parts of the dialogs in order to concentrate on the effects of MT by circumventing communication problems caused by speech recognition errors (Takezawa and Kikui, 2003). In this case, the resulting system is considered equivalent to using an S2ST system whose speech recognition part is almost perfect. The environment with human typists is somewhere between the "Wizard of Oz" approach used in Verbmobil (Jekat and Hahn, 2000) and in creating the SLDB, which replaced the entire S2ST process with humans, and an approach that relies only on an S2ST system (Costantini et al., 2002). In order to investigate the effects of including speech recognizers, we substituted real speech recognizers for human interpreters in some dialogs. An overview of the data collection environment is shown in Figure 10.18.

Translation quality generally depends on the linguistic and acoustic properties of the training and test corpora. We have carried out five sets of simulated dialogs so far, changing the complexity of the task that the speakers needed to carry out and the instructions given to the speakers.

We divided the dialog tasks into three classes: simple, medium, and complex. A task in the simple class consisted of a single request to which a simple answer was expected, such as "asking an unknown foreigner where a bus stop to downtown is." The medium class tasks included two or three negotiations, each requiring two or three utterance turns, such as ordering meals at a restaurant by asking today's special. A task in the complex

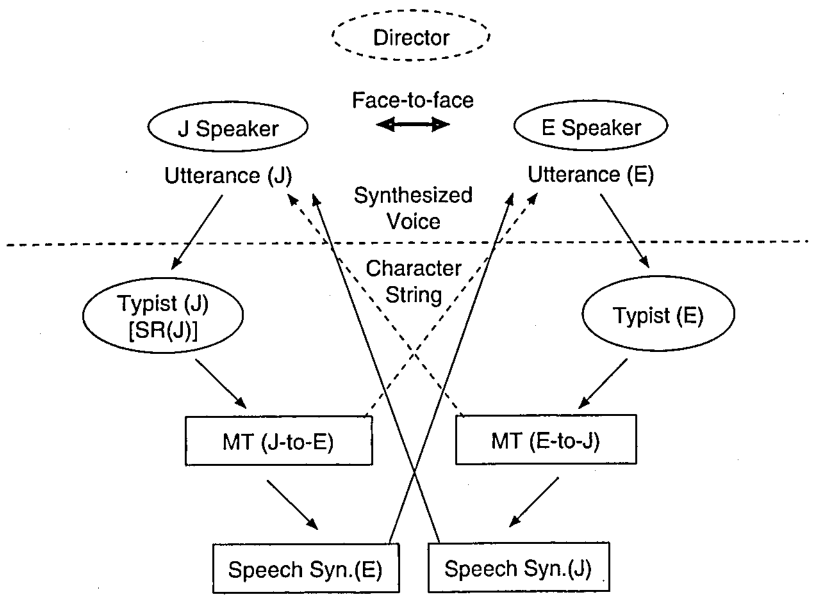


Figure 10.18: Data-collection environment of MAD.

class consisted of mutually dependent sequences of information exchange. A typical example is planning a guided tour through a conversation with a travel agent.

The MAD corpus consists of five sets of dialogs. The first set of dialogs (MAD1) was collected to see whether conversation through a machine translation system is feasible. The second set (MAD2) focused on task achievement by assigning complex tasks to participants. The third set (MAD3) contained carefully recorded speech data using medium complexity. The fourth set (MAD4) aimed at investigating the relations between instructions and utterances. For this set, we gave different types of instructions before starting the dialog. The least restrictive instruction was something like "Speak loudly and clearly," whereas a more restrictive one was "Try to divide utterances into sentences that include one topic in one sentence." For the final set (MAD5), we controlled extra information given to speakers, such as the translated text displayed on a PDA, to see whether the information affected the dialogs.

Table 10.11 shows a summary of the five experiments, MAD1–MAD5. In this table, the number of utterances includes both Japanese and English.

Table 10.11 Statistics of MAD corpora.

Subset ID	MAD1	MAD2	MAD3	MAD4	MAD5
# of utterances	3022	1696	2180	1872	1437
# of morphs per utterance	10.0	12.6	11.1	9.82	8.47
# of utterances per dialog	7.8	49.3	18.8	22.0	27.0
Task Complexity	Low	High	Medium	Medium	Medium

Average numbers depend on experimental conditions

10.4.3 Component Evaluation

Japanese Speech Recognition

The ML-SSS algorithm and the MDL-SSS algorithm were compared to create Japanese context-dependent HMMs. As described in the previous sections, the resulting acoustic models can capture both contextual and temporal variations, while decision-tree clustering can capture contextual variations only. Furthermore, the MDL-SSS algorithm can estimate the best model almost automatically, while the ML-SSS algorithm needs to find the best parameters via experiments.

For the acoustic training set, we used the Japanese travel dialogs in "The Travel Arrangement Task (TRA)." We also used 503 phonetically balanced sentences (BLA) read by the same 407 speakers of the TRA. TRA includes about 5 hours of speech, and BLA includes about 25 hours of speech. The TRA corpus includes many expressions that are similar to those of the BTEC, and the BLA corpus is helpful for creating Japanese standard phoneme models.

The analysis conditions were as follows: The frame length was 20 ms and the frame shift was 10 ms; 12-order MFCC, 12-order ΔMFCC, and Δ log power were used as feature parameters. The cepstrum mean subtraction was applied to each utterance. We used 26 Japanese phonemes and silence. Table 10.12 shows the phoneme units for the Japanese ASR.

Table 10.12 Phoneme units for Japanese ASR.

Vowels	a, i, u, e, o
Consonants	b, ch, d, g, f, h, j, k, m, n, ng, p, q, r, s, sh, t, ts, w, z, zh

The silence model with three states was built separately from the phoneme models. Three states were used as the initial model for each phoneme. The scaling factors $C_c = 2, C_t = 20$ were used for the MDL-SSS. After a topology was obtained by each topology training method, mixture components were increased, and a five Gaussian mixture model was created.

The transcriptions of the BTEC and SLDB corpus were used to create language models. The training databases included 6.2 million words. A word bigram model, a word trigram model, and a Multi-Class Composite (MCC) bigram model were created. The MCC bigram model included 4,000 classes for each direction. The size of the lexicon was 54,000 words, and the number of extracted composite words was 24,000 words. For recognition, the gender-dependent acoustic model and the MCC bigram model were used in the first pass, and the word trigram model was used to rescore word lattices in the second pass.

For the test set, the Japanese test set 01 of the BTEC was used, which contains 510 sentences. As a speech database, we collected utterances by 20 males and 20 females. Each speaker uttered 102 utterances included in the test set 01.

As Table 10.13 shows, the perplexity of the MCC bigram model lies between the word bigram and the word trigram model.

Table 10.14 shows the recognition performance represented by word accuracy (WA) rates for the Japanese BTEC test set 01. The model with 2,086 states, created by the MDL-SSS, using scaling factors $C_c = 2$,

Table 10.13 Perplexity for Japanese BTEC test set 01.

word 2-gram	word 3-gram	MCC 2-gram
30.64	17.45	24.81

Table 10.14 Recognition performance for Japanese BTEC test set 01.

	#states	WA[%]
ML-SSS	2,100 (max state length = 4)	94.51
MDL-SSS	2,086 ($C_c = 2, C_t = 20$)	94.41

Table 10.15 Word accuracy rates [%] for two different language model combinations (the MDL-SSS acoustic model).

Model	Word 2-gram		MCC 2-gram	
3-gram Rescore	No	Yes	No	Yes
Acc.(%)	91.98	93.84	93.37	94.41

$C_t = 20$, obtained almost the same performance as that with 2,100 states created by the ML-SSS.

Table 10.15 shows the word accuracy rates by two combinations of language models. For the first-pass search, one used the word bigram model and the other used the MCC bigram model. Furthermore, both of them used the word trigram model for rescoring in the second pass search. The acoustic model was the same as the MDL-SSS model with scaling factors $C_c = 2, C_t = 20$ in Table 10.14. The MCC bigram model obtained a 17.3% error reduction rate compared to the word bigram model, and the combination of the MCC bigram model and the word trigram model obtained a 9.25% error reduction rate compared to the combination of the word bigram model and the word trigram model.

English Speech Recognition

In contrast to the Japanese speech recognition system, in-domain acoustic training data were not available at the first stage of developing the acoustic model of the English speech recognition system. However, as Lefevre et al. (2001) demonstrated, out-of-domain speech training data do not cause significant degradation of system performance. In fact, it was found to be more sensitive to the language model domain mismatch. Thus, we chose the *Wall Street Journal* (WSJ) corpus for acoustic model training, since we needed a speech database that was large enough and that contained clean speech from many speakers. About 37,500 utterances recommended for speaker-independent training (WSJ-284) were selected as the training set for our acoustic model. The total number of speakers was 284 (143 male and 141 female). Feature extraction parameters were the same as for the Japanese language system: 25 dimensional vectors (12 MFCC + 12 Delta MFCC + Delta pow) extracted from 20 ms long windows with a 10 ms shift. First, we trained a model of the same size and topology with the same training method as the Japanese baseline—that is, 1,400 states with five mixture components per state and the ML-SSS algorithm. This was rather small compared to the other models that have been built on the same data (IWSLT workshop 1994), so it was not expected to have high performance. Nevertheless, we regarded it as a starting point for further model development and optimization. Next, we trained several models using the MDL-SSS algorithm, in which the temporal splitting constant C_t was set to 20 and the contextual splitting constant C_c took values from 2 to 10. In this way, we obtained models with state numbers ranging from about 1,500 to about 7,000. Initially, they all had five mixture components per state. The preliminary tests showed that the model with 2,009 states was the best and was therefore selected for further experiments. Two more versions of this model—with 10 and 15 mixture components per state—were trained as well.

The language model training data consisted of 600,000 English sentences of BTEC and SLDB and about 3.4 million words. Standard bigram and trigram models were trained as well as one MCC word bigram model. The number of classes was 8,000, while the number of composite words was about 4,000.

Although the BTEC task domain is quite broad, there are many travel-oriented words that are not included in publicly available pronunciation dictionaries. Also, there are many specific proper names of sightseeing places, restaurants, travel-related companies, and brand names. A large portion of the task word list represents Japanese words, including Japanese first and family names. In total, there were about 2,500 such words ($\approx$10% of the 27,000-word dictionary), and to develop good pronunciation variants for them was quite a challenge. For Japanese words, because there is no principled way to predict how a native English speaker would pronounce them, the pronunciation will depend heavily on the speaker's proficiency in Japanese, ranging from being fluent to speaking just a couple of widely known words. Therefore, we decided to mimic the first extreme by taking one pronunciation variant from the Japanese dictionary and converting the Japanese to the English phone set, and to mimic the second extreme by generating another pronunciation variant by following the English grapheme-to-phoneme rules. The latter was done by using the TTS software "Festival" followed by a manual correction of some of the pronunciations judged as "making no sense."

The English phoneme set consisted of 44 phonemes, including silence. They were the same as those used in the WSJ corpus official evaluations

Table 10.16 Acoustic model performance comparison.

Model	ML-SSS	MDL-SSS			
State #	1,400	1,578	2,009		3,028
Mix. #	5	5	5	15	5
Acc.(%)	87.5	88.1	88.5	89.4	88.2

Table 10.17 Language model performance comparison.

Model	Word 2-gram		MCC 2-gram	
3-gram Rescore	No	Yes	No	Yes
Acc.(%)	89.21	92.35	89.63	93.29

because in this way, we could use its dictionary as a source of pronunciation base forms. In addition, we could run the WSJ task tests with our model to compare performance.

In the first series of experiments, we evaluated the performance of the several acoustic models we have trained. The test data comprised 1,200 sentences from 35 speakers. Small conventional bigram and trigram language models covering about 25% of all text training data were used to speed up the evaluation. The recognition results in terms of word accuracy are given in Table 10.16. As can be seen, the MDL-SSS model with 2,009 states and 15 mixture components was the best one, thus it was used for the next experiments involving different types of language models.

Next, we evaluated the language model's performance. In these experiments, we used 204 utterances taken randomly from the larger BTEC test set. The results are summarized in Table 10.17.

Chinese Speech Recognition

The basic subword units used for the Chinese speech recognition front end were the traditional 21 initials and 37 finals, as illustrated in Table 10.18.

The acoustic model was developed using a well-designed speech database: the ATR Putonghua (ATRPTH) speech database of 2003 (Zhang et al., 2003). This database has a rich coverage of the triplet initial/finals

Table 10.18 Subword units for Chinese ASR system.

Unit	Types
Initials	b, p, m, f, d, t, n, l, g, k, h, j, q, x, z, c, s, zh, ch, sh, r
Finals	a, ai, an, ang, ao, e, ei, en, eng, er, i1, i2, i3, ia, ian iang, iao, ie, ing, in, iu, iong, o, ou, u, ua, uai, uang uan, ui, un, uo, ong, v, van, ve, vn

Table 10.19 Token coverage rates of different subword units.

Unit	792 Set	Newspaper	Token Coverage
A	974	1,306	98.81%
B	402	408	99.99%
C	10,906	48,392	70.15%
D	4,653	4,598	99.42%

phonetic context, and sufficient samples for each triplet with respect to balanced speaker factors, including gender and age.

The phonetically rich sentence set of ATRPTH has 792 sentences. An investigation on the *token coverage rates* has been carried out on a one-month volume of daily newspapers for different types of phonetic units. Table 10.19 shows the results, where

- Unit A: represents the tonal syllable
- Unit B: represents the base syllable without tone discrimination
- Unit C: represents the normal initial/final triplets
- Unit D: represents the context-tying initial/final triplets, which are tied based on phonetically articulatory configurations, and are assumed to cover the major variants of each triplet phonetic context (Zhang et al., 2001).

The speakers were chosen to have a balanced coverage of different genders and ages. Each unique triplet has at least 46 tokens in the speech database, guaranteeing a sufficient estimation for each triplet HMM.

During the model estimation, accurate pause segmentation and context dependent modeling were done iteratively to guarantee the model's accuracy and robustness (Zhang et al., 2003a). The HMM structure was derived

Table 10.20 Chinese character based recognition performance.

Group	Character Corr.	Character Acc.
Male	96.1%	95.7%
Female	95.2%	94.4%
Total	95.7%	95.1%

through a phonetic decision tree based maximum likelihood state splitting algorithm. The acoustic feature vector consisted of 25 dimensions: 12 dimensional MFCCs, their first order deltas, and the delta of frame power. The baseline gender-dependent HMM had 1,200 states, with 5 Gaussian mixtures in each state.

The language model for Chinese ASR also used the composite MCC N-gram model. The basic lexicon had 19,191 words, while the BTEC Chinese corpus contained 200,000 sentences for LM training. After they were segmented and POS tagged, word clustering was investigated based on the right- and left-context Markovian dependencies. A normal word based bigram model showed a perplexity of 38.4 for the test set with 1,500 sentences. With a clustering of 12,000 word classes, the MCC bigram model showed a perplexity of 34.8 for the same test data. The bigram language model was used to generate a word lattice in the first pass, and a trigram language model with a perplexity of 15.7 was used to rescore the word lattice.

The evaluation data were the BTEC Chinese language-parallel test data, which included 11.59 hours of speech by 20 females and 20 males. The ages of the speakers ranged from 18 to 55 years. All the speakers spoke Chinese Putonghua, with some accent.

Table 10.20 shows the gender-dependent, Chinese character based recognition performances. The total performance was 95.1% for Chinese character accuracy with a real-time factor of 26. The performance degraded to 93.4% when the search beam was narrowed to obtain a real-time factor of 6, emphasizing the need for algorithms to increase the search speed without sacrificing performance.

Text-to-Speech Evaluation

A perception experiment was conducted in which the naturalness of synthetic speech for XIMERA for Japanese and 10 commercial TTS systems

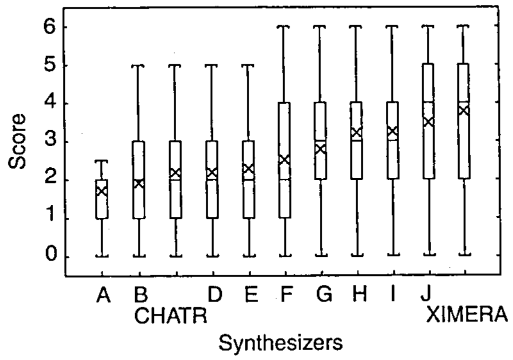


Figure 10.19: The result of an evaluation experiment for naturalness between several TTS products. The horizontal bars at the top, middle, and bottom of the boxes indicate 75%, 50%, and 25% quartiles. The vertical lines extend to the closer point of either the maximum/minimum value or 1.5 times the interquartile distance. Mean values are indicated by "x" marks.

was evaluated. A set of 100 Japanese sentences that were evenly taken from 10 genres was processed by the 11 TTS systems to form a set of stimuli comprising 1,100 synthetic speech samples. The stimuli were randomized and presented to 40 listeners through headphones in a quiet meeting room. The listeners rated the naturalness of each stimulus on a 7-point scale from 0 (very bad) to 6 (very good).

Figure 10.19 shows the result, in which XIMERA outperforms the other systems. However, the advantage over the second-best system, which is not a corpus-based system, is not substantial, although it is statistically significant.

A perception experiment for naturalness of synthetic speech for Chinese XIMERA is undergoing planning, and we have no evaluation results comparing other Chinese TTS systems.

10.4.4 Machine Translation Evaluation

Rank Evaluation Method

The rank evaluation methods are the simplest and the most common among test-set-based evaluations. The training corpora used for the machine

Table 10.21 Translation quality of four systems for BTEC.

	SAT	HPAT	D-cube	SELECTOR
A	67.2549	42.5490	63.7255	68.2353
AB	74.7059	63.7255	72.1569	75.8824
ABC	82.5490	79.0196	78.8235	83.5294

translation systems were the BTEC, described in Section 10.4.2. The test set consisted of 510 pairs randomly selected from BTEC and kept unused for training. The target part of the test set consisted of the paraphrasing of up to sixteen multiple reference translations for each source sentence, which are utilized for automatic evaluation programs.

Translations by four machine translation systems—SAT, HPAT, D-cube, and SELECTOR—were shown simultaneously to several native English professional interpreters. The evaluation was done according to ATR's evaluation standard of four grades: (A) Perfect: no problems in either information or grammar; (B) Good: easy to understand, with either some unimportant information missing or flawed grammar; (C) Fair: broken but understandable with effort; and (D) Nonsense or No-output. Each translation was finally assigned to the median grade from among its grades from multiple evaluators.

Table 10.21 shows the translation quality of Japanese to English translations for BTEC. The figures are accumulative percentages for the quality grade. It is fairly high even for the difficult language pair of Japanese to English. In addition, we can see in every grade, A, A+B, A+B+C, that the SELECTOR outperforms every single element machine translation.

Translation Paired Comparison Method

When considering the users' viewpoint, it is ideal to assemble people with various levels of skills in foreign languages and to ask them to evaluate their satisfaction with the achievement of the dialog through the system. This is important because they are thought to be influenced by their target language skill. However, experimental costs are so prohibitive that dialog experiments have never been carried out except evaluation tests for the final stage of development. To address this problem, Sugaya et al. (2000) proposed a translation paired comparison method that can precisely evaluate

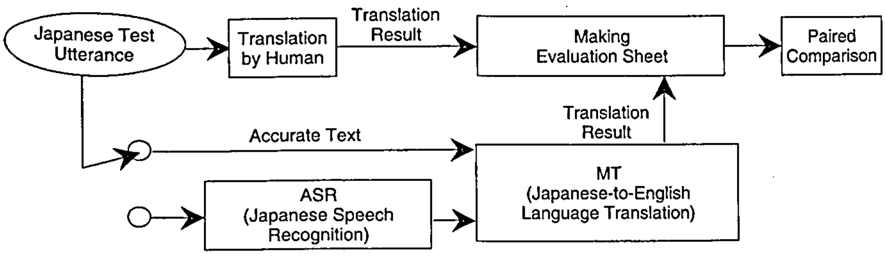


Figure 10.20: Diagram of translation paired comparison method.

systems' performance. For this method, the evaluation results are given as the systems' TOEIC scores. TOEIC is an acronym for **Test of English for International Communication**, which is a test for measuring the English proficiency of non-native speakers, such as Japanese (TOEIC, 2005). The total score ranges from 10 (lowest) to 990 (highest).

Figure 10.20 shows a diagram of the translation paired comparison method in Japanese-to-English translation. Here, Japanese native-speaking examinees were asked to listen to spoken Japanese text and then write an English translation. The Japanese utterance was presented twice within one minute, with a pause between the presentations. To measure the English capability of the examinees, their TOEIC scores were used. The examinees were asked to present their official certificates showing the TOEIC score they had earned on the test within the past six months.

In the translation paired comparison method, translations by the examinees and the output of the system were printed together in rows with the original Japanese text to form evaluation sheets for comparison by a bilingual evaluator. Each transcribed utterance on the evaluation sheets was represented by the Japanese test text and the two translation results.

The evaluator followed the procedure depicted in Figure 10.21. Ranks in the figure were defined as follows: (A) Perfect: no problem in both information and grammar; (B) Fair: easy-to-understand with some unimportant information missing or flawed grammar; (C) Acceptable: broken but understandable with effort; (D) Nonsense: important information has been translated incorrectly.

In the evaluation process, the human evaluator ignored misspellings because the capability being measured was not English writing but speech translation. From the scores based on these rankings, either the examinee or

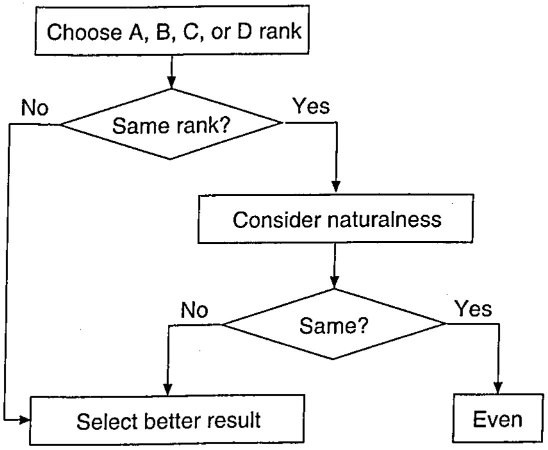


Figure 10.21: Procedure of comparison by bilingual evaluator.

the system was considered the "winner" for each utterance. If the ranking and the naturalness were the same for an utterance, the competition was considered "even."

To prepare for regression analysis, the number of "even" utterances were divided in half and equally assigned as system-won utterances and human-won utterances. Accordingly, we define the human winning rate W_H by the following equation:

$$W_H = (N_{human} - 0.5 \times N_{even})/N_{total}, \qquad (10.11)$$

where N_{total} denotes the total number of utterances in a test set, N_{human} represents the number of human-won utterances, and N_{even} indicates the number of even (nonwinner) utterances—that is, no quality difference between the results of the system and those of humans.

In the regression analysis, we regarded the examinees' TOEIC score as an independent variable and W_H as a dependent variable. In other words, W_H (Y_i) and the TOEIC scores for the examinees (X_i) are assumed to satisfy the population regression equation:

$$Y_i = \beta_1 + \beta_2 X_i + \epsilon_i \qquad (i = 1, 2, \ldots, m), \qquad (10.12)$$

where β_1 and β_2 are population regression coefficients and m is the number of examinees.

"System's TOEIC score" defines the capability-balanced point between the system and the examinees, and it can be determined as the point at which the regression line crosses half the total number of test utterances, that is, W_H of 0.5.

Figure 10.22 shows an example of evaluation results. In this case, the total number of examinees was 29, and their TOEIC scores ranged from 300 to 800. Every hundred-point range had five examinees, except for the 600s, which had four examinees. The test set consisted of 330 utterances from the SLDB corpus. In this figure, the straight line indicates the regression line. This system's TOEIC score is 705, the cross point of the regression line with $W_H = 0.5$. Consequently, the translation capability of the language translation system equals that of an examinee with a score of around 700 points on the TOEIC scale.

Automatic Evaluation Scheme

The translation paired comparison method can precisely evaluate the system's performance, but still leaves the problem of evaluation cost. Therefore, we developed an automatic evaluation scheme that can estimate the system's TOEIC score with an objective method. Basically, the method substitutes the human evaluation process with an objective evaluation method.

The first point to explain is the automation of the translation paired comparison method. There are two levels for applying an objective evaluation method to the automation of the translation paired comparison method: the utterance level and the test set level. However, both BLEU and NIST are inadequate for determining the relative merits at the utterance level. Consequently, we employed evaluation at the test set level, which consists of several hundred utterances. In a sense, this evaluation follows a different procedure from the original translation paired comparison method because the original method compared pairs at the utterance level.

The flow of the evaluation is shown in Figure 10.23. First, we apply the objective evaluation at the test set level to the evaluation of all the examinees' and the system's translations. Then, we carry out a regression analysis using the objective scores. In the regression analysis, we use the objective scores instead of W_H. More specifically, scores by the objective evaluation method (Y_{auto_i}) and the TOEIC scores for the examinees (X_i)

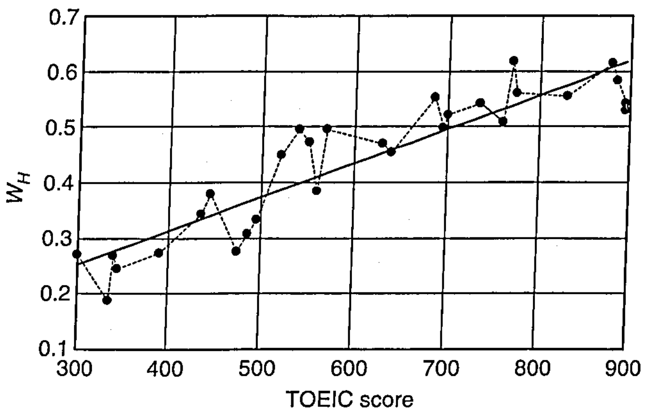


Figure 10.22: Evaluation result with the translation paired comparison method.

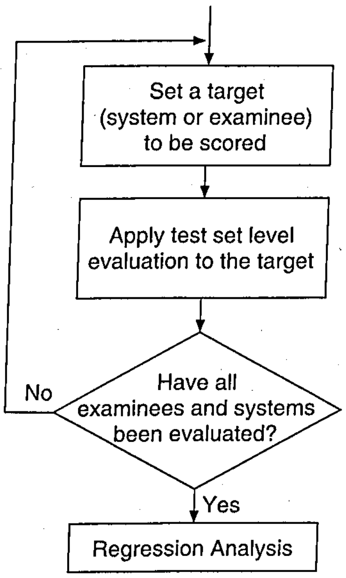

Set a target (system or examinee) to be scored
Apply test set level evaluation to the target
Have all examinees and systems been evaluated?
No
Yes
Regression Analysis

Figure 10.23: Procedure of the automatic evaluation method.

are assumed to satisfy the population regression equation:

$$Y_{auto_i} = \beta_{auto_1} + \beta_{auto_2} X_i + \epsilon_{auto_i} \qquad (i = 1, 2, \ldots, m), \qquad (10.13)$$

where β_{auto_1} and β_{auto_2} are population regression coefficients and m is the number of examinees.

In the automatic evaluation method, the system's TOEIC score is determined to be the point at which the regression line crosses the objective score evaluating the target system.

10.4.5 Overall Evaluation

Overall performance evaluation of the S2ST system was conducted with two modes: a laboratory test conducted under various controlled conditions, and a conversation test in real environments.

Laboratory Test

Translation performance was evaluated on two sets, BTEC1 and MAD4. Evaluation measures were subjective ranking, BLEU (Papineni et al., 2002), mWER (word error rate using multiple references), and the system's TOEIC score with the automatic evaluation scheme. Table 10.22 shows some of the evaluation results. In this table, subjective evaluation sections show the percentage of A, A+B, and A+B+C relative to the size of the test set.

Since BTEC originated from texts, we produced speech by reading them. In this sense, the resulting speech was artificial. In the case of MAD, we applied our speech recognizer to the recorded speech after the data collection. "MT only" for MAD4 shows the result of MT applied to correct transcriptions created manually. Speech recognition results show that we have relatively good results (i.e., just below 90% accuracy) for MAD because people spoke in a system-friendly way as described earlier. When we focus on translation performance, it is clear that BTEC is much easier than MAD. This is partly because the test sets share the same mother corpora with the training corpora. As shown in Table 10.9, BTEC contains many duplicates (e.g., 37% in Japanese) among sentences in the source and

Table 10.22 Translation performance.

Evaluation Set	BTEC1	MAD4
Speech Recognition (SR)		
Word Accuracy (%)	(94.8)	89.5
Sentence Perfect (%)	(81.4)	54.2
MT Only		
Subj (A/AB/ABC)	66.2/77.1/84.7	33.5/50.4/65.7
BLEU	0.59	0.48
mWER	0.31	0.45
SR+MT		
Subj (A/AB/ABC)	-	30.3/45.4/61.8
BLEU	-	0.37
mWER	-	0.54

target languages. Thus, some test utterances happen to be the same as training sentences even though we randomly separated the original BTEC into test and training sets. These "duplicate" utterances do not mean "closed" data since reference translations for the test-set utterances were created independent of original translations in the corpus. In order to know this effect, we divided our test data into those that were included in the training corpus (MATCHED) and the remaining data (UNMATCHED). These two subsets were applied to our MT system.

The results are shown in Table 10.23. We see that the subjective scores of the nonduplicate part are still better than the average scores of MAD. We conclude that BTEC is easier than MAD for the MT module based on the subjective evaluation scores.

Table 10.23 Translation performance for BTEC with/without duplications.

	UNMATCHED	MATCHED
Subj (A/AB/ABC)	43.9/60.4/73.7	88.6/93.7/95.7
BLEU	0.51	0.73
mWER	0.42	0.19
Perplexity	36.3	14.1

Table 10.24 Translation performance of test set without duplications.

	Low	Middle	High
Subj (A/AB/ABC)	62.3/78.4/89.8	23.4/40.7/62.9	14.9/32.1/44.6
BLEU	0.60	0.49	0.4
mWER	0.31	0.45	0.57
Perplexity	9.8	29.9	97.0
Length*	9.6	12.0	12.4

**Length = # of words per utterance*

To have a closer look at MAD, we divided the test utterances into three subsets with the same size in terms of their perplexity[2] and calculated scores for each subset, as shown in Table 10.24. The table shows that the perplexity of each subset is clearly different and that the translation performance negatively correlates with the perplexity. Consequently, we need to improve the translation performance of mainly high-perplexity utterances.

The system's TOEIC scores with the automatic evaluation scheme were also obtained. The system achieved on the BTEC and MAD subsets of low and middle perplexity a TOEIC score of more than 700. This is 50 points higher than the average score of a Japanese businessperson in the overseas departments of many Japanese corporations. On the contrary, the system's TOEIC score for all the test sets including the BTEC and every subset of MAD decreases to about 500 due to low TOEIC score for the subset of MAD with high perplexity, which is mainly due to low recognition accuracy, and a loss of fluency in the translated expressions.

These results support the bootstrapping style approach. Another promising approach is to create corpora that align high-perplexity utterances with their simplified expressions (i.e., paraphrases) and develop a corpus-based paraphrasing system (Shimohata et al., 2004). For this purpose, simplified utterances in MAD can be used as communication oriented paraphrases for S2ST.

Evaluation Tests in Real Environments

Real environments have a complex and often uncontrollable impact on S2ST systems. There is a limit to carrying out such evaluation tests with

[2]Perplexity was calculated in the source language.

regard to the cost of conversation experiments in real environments, however, evaluation tests in real environments are important for fully evaluating the availability of the system. Therefore, we conducted an evaluation test of S2ST systems from the viewpoint of user satisfaction in locations where S2ST systems may be necessary, such as international airports.

Prototype S2ST System

ATR has developed two kinds of prototype S2ST systems for conducting evaluation tests in real environments—one between English and Japanese, and the other between Chinese and Japanese. These prototype systems consist of two sets of handheld PDAs (Personal Digital Assistants) connected by means of wireless LAN to server modules of speech recognition, machine translation, and speech synthesis in the network. One PDA is used as an input device for Japanese and an output device for Japanese translated from English or Chinese. The other is used as an input device for English (Chinese) and an output device for English (Chinese) translated from Japanese. In this configuration, two sets of PDAs are used only as speech input and output devices and for some additional functions, such as speech detection, filtering, and noise reduction. We employed this configuration because we believe that (1) an input/output device should be portable and as light as possible in order to be taken everywhere, and (2) S2ST systems should accept expressions that are as diverse as possible to support communications in various situations. Since the ambient noise at the evaluation location (airport) is very high and nonstationary, we used head-mounted close-talking microphones.

Evaluation Test Procedure

The evaluation test sites were installed near the tourist information desk in the Kansai International Airport and downtown in Osaka city. The speakers of Japanese were staff members of the tourist information desks, who frequently speak English but not Chinese. The speakers of English and Chinese were tourists walking in the airport or downtown who volunteered to participate in the experiment.

After a short introduction on how to use the PDA, they were asked to do three kinds of tasks. In one, they were asked to present some predetermined simple questions, such as "Where is the taxi stand?" to the staff members. In the second, they were asked to make conversation consisting of two or three turns, such as how to go to downtown. In the third, the speakers were requested to ask some questions about their travel, which is the most difficult task for S2ST systems. After conversations using the S2ST systems, they were requested to fill in a questionnaire. We conducted the evaluation tests for three days at each site, with about fifty subjects each for English and Chinese.

Evaluation Results

Figure 10.24 shows the evaluation results based on two questionnaires: one about the user's impression on the success of communication, and one about response time. As can be seen, more than 70% of English subjects think that a conversation could be carried out (almost) successfully, while only 60% of the Chinese subjects believe this. The performance degradation of the Chinese S2ST may be due to (1) the smaller volume of the Chinese-Japanese BTEC corpus, which degrades the performance of the machine translation module between Chinese and Japanese, and (2) the lower accuracy of the Chinese speech recognition module, which also results from the smaller volume of the Chinese speech database.

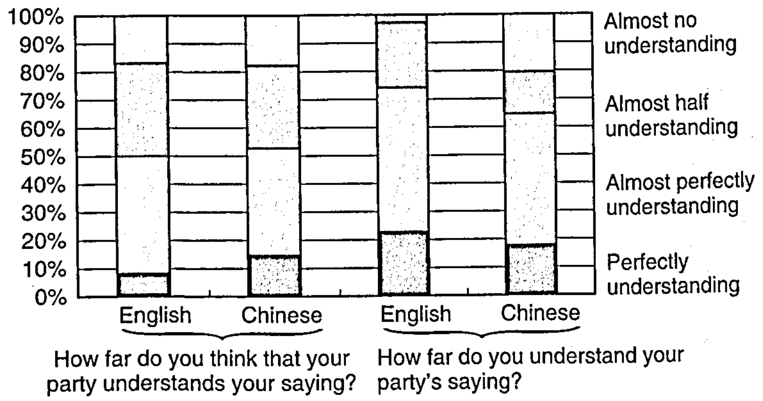


Figure 10.24: Evaluation results in real environments.

The evaluation results indicate that state-of-the-art S2ST systems evolved into a stage that allows communication across language boundaries, however, accuracy and response time (not shown in Figure 10.24) need further improvement.

10.5 Conclusion

10.5.1 Speech Translation Strategies

We investigated different translation strategies: Interlingua-based translation, statistical mapping into an interlingua representation, direct statistical translation, and using English as a pivot language. Designing an interlingua and writing the analysis and generation grammar is time consuming and requires highly trained linguists. We further introduced a statistical interlingua-based approach that applies techniques that have been initially developed for mapping word sequences into tree structures in direct statistical MT and have been extended for this purpose. This still requires the design of the interlingua and the annotation of sufficient data with the interlingua, but replaces the manual writing of grammars by automatically learning the mapping of the source sentence into the interlingua representation. In our experiments, this statistical approach to interlingua-based translation did not perform as well as the manually crafted grammars. To some extent this is due to data-sparseness problems. However, the manual grammars include some phrasal translations, which improve translation quality. Such a translation memory mechanism, mapping entire sentences to the appropriate interlingua representation, could be added to the statistical IL system as well to get further improvements.

We also investigated the performance of a direct statistical translation system, which is based on word-to-word and phrase-to-phrase alignments trained from the same data. Despite the general belief that statistical machine-translation systems can only work when large bilingual corpora are available, the direct statistical system outperformed the grammar-based system. As the statistical system used only the translations, this development cost is significantly lower than that for the interlingua system. And the statistical system is flexible in that additional data, like available dictionaries or additional monolingual data to train the language model, can be easily added to improve the performance.

The comparison of translation approaches suggest that MT systems can be successfully constructed for any language pair by cascading multiple MT systems via English. Moreover, end-to-end performance can be improved if the interlingua language is enriched with additional linguistic information that can be derived automatically and monolingually in a data-driven fashion.

When dealing with speech translation, we are faced with disfluencies and with errors from the speech recognizer. To handle disfluencies, we developed a consolidation module, which detects and removes these disfluencies. The approach is based on a noisy channel approach, borrowing essentially from the statistical machine-translation techniques.

Finally, we investigated better ways to couple speech recognition and translation to improve translation quality by optimizing the overall system. Translating all the paths in the word lattice generated by the speech recognition system and using the acoustic scores in addition to the translation and language model scores resulted in an improvement over translating only the first-best recognizer output.

Much remains to be done to bring robust speech translation to practical day-to-day use in the many languages of the world. Our experiments indicate that data-driven approaches—automatically learning from bilingual corpora—is the most competitive approach to rapid building of speech translation systems. So far, systems have been demonstrated for limited domain speech translation tasks, and these will remain important in the future. However, steps should and, we believe, can be taken now toward domain-unlimited speech translation.

10.5.2 Portable Speech-to-Speech Translation

We have conducted research on corpus-based technologies because we believe that corpus-based technologies are suitable for S2ST, taking into consideration the points of (1) multilanguage systems, (2) domain portability, and (3) the technology trend of each component technology for S2ST. In order to develop S2ST technologies, we have created various speech databases for speech recognition and speech synthesis, and also three different types of corpora in the travel domain: (1) a large-scale multilingual collection of basic sentences called BTEC, (2) a small-scale bilingual collection of spoken sentences called SLDB, and (3) a small-scale corpus of the MT-assisted dialogs called MAD.

We have developed two kinds of corpus-based S2ST systems based on various machine-learning algorithms for each component, a huge speech database, and multilingual corpora. One is the S2ST system between English and Japanese; the other, between Chinese and Japanese. We have conducted various evaluation tests of the S2ST systems using a subset of the BTEC and MAD. Thanks to many various technologies in each component module, and many large speech and text corpora, the S2ST systems provide good translation quality for utterances of the BTEC and also for utterances of the MAD with low and middle perplexity. We have also conducted evaluations of the systems from the viewpoint of user satisfaction in real environments, such as an international airport. The evaluation results show that the performance of the S2ST systems is not fully satisfactory, however, 60–70% of users think that conversation could be carried on successfully or almost successfully using the S2ST systems. One of the most important results is that about 60% of the subjects thought that the S2ST systems could be applicable to communication between people whose mother tongues are not understandable to each other.

However, the state-of-the-art S2ST technology still has a lot room for improvement. One of the problems to be resolved is to shorten the response time, especially for long utterances. It is desirable from the users' viewpoint to provide confidence measures for translated sentences. As is shown in the section on automatic evaluation schemes, we have an objective evaluation measure for estimating the quality of translated sentences in the test-set level, which shows the average performance for collection of utterances; however, we do not have adequate measure for estimating the quality in the utterance level. The other point is how to exchange information among each module. We have employed a simple cascade configuration of the speech recognizer and machine translation, in which the single best output of the former module is fed into the latter. The N-best lists of the former output contain the better recognition result, which may produce better translation results. In order to improve the performance of the S2ST system, it is important to investigate ways of improving the performance of each module but also to achieve more collaborative functions between the speech recognition module and the machine-translation module. One way is to develop larger multilingual corpora, which cover more domains and diverse expressions. A promising approach is to create corpora that align high-perplexity utterances with their simplified expressions (i.e., paraphrases) and develop a corpus-based paraphrasing system (Shimohata et al., 2004). For this purpose, simplified utterances in MAD can be used as communication-oriented paraphrases for S2ST.