Connectionist Transfer in Machine Translation

Ye-Yi Wang and Alex Waibel
Interactive Systems Laboratories
Carnegie Mellon University
Pittsburgh, PA 15213
yyw@cs.cmu.edu, waibel@cs.cmu.edu

Abstract

A traditional transfer system in machine translation maps between language structures and an intermediate representation. Our connectionist transfer system maps from f-structures of one language to f-structures of another language. It encodes the intermediate representation implicitly in neural networks' activation patterns. The system is learnable, therefore it does not need any effort in hand-crafting the representation and mapping rules. Experiments show the system has good scalability and generalizability performance.

1 Introduction

Most of the current machine translation systems adopt an indirect strategy that maps between languages and an intermediate representation. The *interlingua* model (Witkam 83; S. Nirenbrug & Tuker 87) uses a language-independent intermediate representation. Design of the representation requires cross-linguistic expertise. The intermediate representation in a *transfer* model (White 87) is language-dependent. Its design is relatively easier. However, multiple such representations are required for a multi-lingual translator. Both models rely upon hand-crafted mapping rules, which demand tremendous human effort.

The difficulties appeal to automatic learning of intermediate representations and mapping rules. (Chrisman 91) proposed a connectionist confluent influence mechanism that acquired the distributed inter-language representation of sentences during its learning to achieve the tight coupling between the representations of sentences in two different languages. The approach was hard to scale up for larger tasks or to generalize for unseen inputs, mostly due to its over-simplified representation of sentences.

In this paper, we present a connectionist mapper. It can learn the transfer from a source language (English) LFG f-structure (Bresnan 82) into its corresponding target language (German) f-structure. It does not need explicit intermediate representation or mapping rules. Instead, the connection patterns of the neural networks implicitly encode the rules and representation.

The domain of our task was the Conference Registration Telephony Conversations. It covered a wide range of topics related to conferences, such as registration, cancellation, hotel reservation, conference information inquiry, etc. Following are some English sentences:

```
  I'll send you the registration form
immediately.
  Can I join the city tour?
  What papers will be presented on
August twenty second?
  When will you check in?
  The evening of May 2nd checking out
the morning of the 8th.
```

The lexicon for the task contained about 400 English and 400 German words in root forms. About 300 pairs of f-structures of the English and German sentences were available from parsers developed in other research tasks (Waibel & etal. 91; Osterholtz & etal. 92).

A machine translation system for the Conference Registration task consisted of three parts: a parser deriving the f-structure from an input source language sentence, a mapper generating a target language f-structure from its source language counterpart, and a text generator producing a target language sentence from its f-structure. According to our experience, mapping between f-structures was the most difficult part, which required the hand-crafting of an intermediate representation and the rules that map between f-structures and the intermediate representation. An automatic transfer system is thus desirable to generate the target language f-structures. The system should have the following properties:

Learnability: The system should be able to learn the structure transfer automatically from paired samples. It should not require hand-crafting of any explicit representations and mapping rules.

Scalability: With limited retraining, the system should be able to deal with larger tasks with an expanded lexicon.

Generalizability: The system should have satisfiable performance on unseen inputs.

2 F-structure Representations

An f-structure is a structured functional representation of a sentence or a phrase. It is composed of a head, several terminal features, and some sub-structures. For the f-structure in [Figure 1(a)], *SEND is the head. The contents in the inner brackets are the sub-structures, whose *grammatical relations* or *roles*[1] are labeled next to the brackets. The rest parts in [Figure 1(a)] are the terminal features. A sub-structure can be referred to with its grammatical relation or its phrasal category (NP, VP, ...). Thus the sub-structure $[_{subj}$ *YOU] can be called either a *SUBJECT sub-structure* or an *NP sub-structure.* The SUBJECT, RECIP and OBJECT sub-structures are the three *immediate sub-structures* of the top level f-structure in [Figure 1(a)], because there is no intervening structure between these sub-structures and the top level f-structure. The DET sub-structure is not an immediate sub-structure of the top level f-structure. Instead, it is an immediate sub-structure of the OBJECT sub-structure. If A is an immediate sub-structure of B, then B is the *parent structure* of A.

A symbolic f-structure cannot be presented to a neural network directly. [Figure 1(c)-(f)] illustrates how an f-structure can be coded as a network's input. Below are the terms used for the representation.

A **lexical vector** is used to code a lexical item. Assuming that every lexical item is an entry in a two-dimensional space instead of a one-dimensional word list, we need two indices to specify the position of a lexical item in the space. Lexical vector is a 0-1 vector with exactly two elements being 1 (being activated). The positions of the two activated elements in the vector specify the two indices for an item in the 2D lexicon[Figure 1(c)][2].

The **terminal feature vector of an f-structure** codes the terminal features of the f-structure. Each element of the vector corresponds to a feature-value pair like (TENSE *PRESENT). The vector, again, is a 0-1 vector with the activated elements indicating that their corresponding feature-value pairs are terminal features of the f-structure [Figure 1(d)]. Since there are altogether around 60 different values for all the features used in the f-structure, the length of the terminal feature vector is around 60.

The **HF-Vector of an F-structure** is the concatenation of the lexical vector of the head and the terminal feature vector of the f-structure [Figure 1(e)].

Thus an f-structure can be represented by its HF-vector and its sub-structures' HF-vectors [Figure 1(f)].

3 The Mapper

A mapper consists of

1. a symbolic controller that assigns an f-structure transfer task to a neural network and interpreting the network's output. According to the interpretation, it recursively assigns the sub-structure transfer tasks to the related networks, and assembles these networks' results to the target f-structure;
2. seven neural networks that map phrasal f-structures between two languages. Each network is constructed for a phrasal category in the target language: IP(sentences), VP, NP, AP, PP, DP(determiners), and MP(miscellaneous, for phrases like "hello", "oh", etc.).

3.1 Phrasal Networks

A phrasal network has four layers: input, feature, hidden, and output layers [Figure 2(a)]. The **input layer** consists of

1. Slots of the HF-vectors for an input f-structure and its context (parent) structure. Each slot corresponds to a fixed role.

   An input f-structure may have sub-structures of arbitrary depth, but the networks must have fixed number of input slots. Therefore we cannot include all sub-structures' HF-vectors in the networks' input. Instead, we "peal off the shell" of an f-structure — only include the HF-vectors of the immediate sub-structures and their immediate sub-structures in turn for the input f-structure, and the HF-vectors of the immediate sub-structures for the context f-structure. Pre-analysis of the samples reveals the possible roles of the sub-structures that can occur at these levels in f-structures for the seven phrasal categories, and slots are then added to the input and feature layers of the corresponding phrasal networks to take as input the HF-vectors of the sub-structures with those possible roles.
2. Grammatical relation of the input source structure in its context[3]. This input is a 0-

[1] *Grammatical relation* is the linguistic term for subjects, objects, etc. We use it interchangeably with the term *role*.

[2] Viewing the lexicon as 2D reduces the length of the vector used to represent a lexical item. If there are n lexical items in a lexicon, we need vectors of length n to represent lexical items in the lexicon when it is viewed as a one-dimensional word list, each with exactly one element being activated. Viewing the lexicon as 2D, the vectors used to represent lexical items only need to be of the length $2\lceil\sqrt{n}\rceil$, with exactly two elements being activated. Of course one may view lexicon as of higher dimensions to further reduce the vector length. However, we found that in this case the network training became much more difficult, if not impossible.

[3] Slot position only indicates the role of sub-structures, not the role of the input structure, since

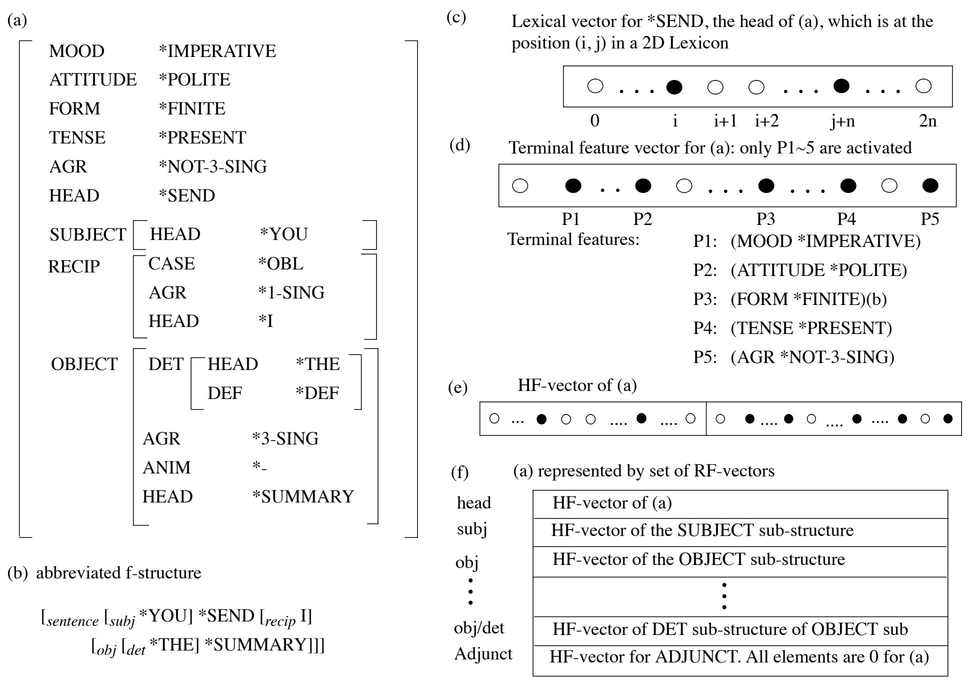
(a)
MOOD *IMPERATIVE
ATTITUDE *POLITE
FORM *FINITE
TENSE *PRESENT
AGR *NOT-3-SING
HEAD *SEND
SUBJECT HEAD *YOU
RECIP CASE *OBL
AGR *1-SING
HEAD *I
OBJECT DET HEAD *THE
DEF *DEF
AGR *3-SING
ANIM *-
HEAD *SUMMARY
(b) abbreviated f-structure
[sentence [subj *YOU] *SEND [recip I] [obj [det *THE] *SUMMARY]]]
(c) Lexical vector for *SEND, the head of (a), which is at the position (i, j) in a 2D Lexicon
0 i i+1 i+2 j+n 2n
(d) Terminal feature vector for (a): only P1~5 are activated
P1 P2 P3 P4 P5
Terminal features:
P1: (MOOD *IMPERATIVE)
P2: (ATTITUDE *POLITE)
P3: (FORM *FINITE)(b)
P4: (TENSE *PRESENT)
P5: (AGR *NOT-3-SING)
(e) HF-vector of (a)
(f) (a) represented by set of RF-vectors
head HF-vector of (a)
subj HF-vector of the SUBJECT sub-structure
obj HF-vector of the OBJECT sub-structure
obj/det HF-vector of DET sub-structure of OBJECT sub
Adjunct HF-vector for ADJUNCT. All elements are 0 for (a)

Figure 1: F-structure Representation. **(a)** an f-structure. **(b)** abbreviation. **(c)**. lexical vector. **(d)**. terminal feature vector. **(e)** HF-vector. **(f)** f-structure represented by HF-vectors.

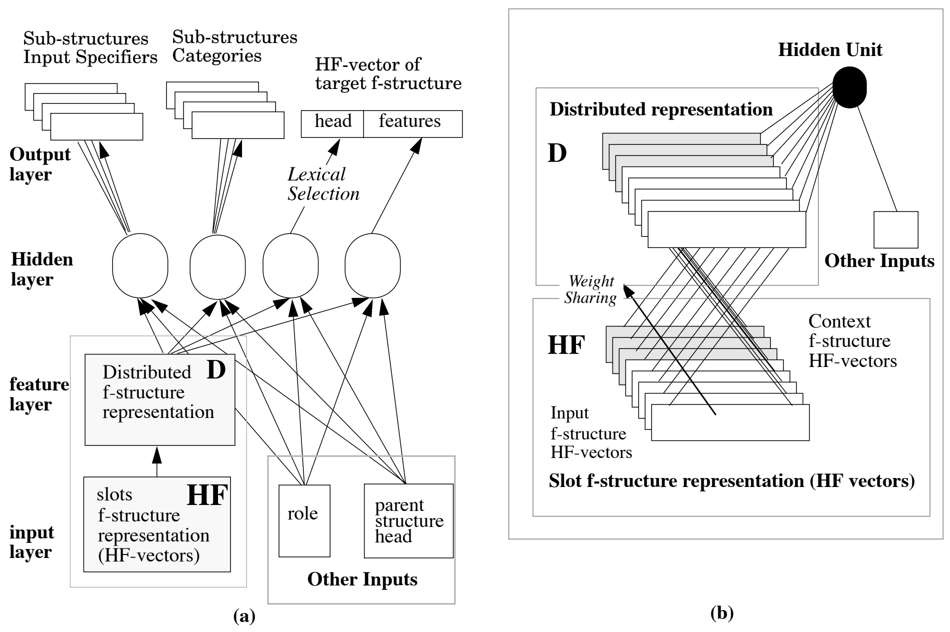
Sub-structures Input Specifiers
Sub-structures Categories
HF-vector of target f-structure
head
features
Output layer
Lexical Selection
Hidden layer
feature layer
Distributed f-structure representation
D
slots f-structure representation (HF-vectors)
HF
input layer
role
parent structure head
Other Inputs
(a)
Hidden Unit
Distributed representation
D
Other Inputs
Weight Sharing
HF
Context f-structure HF-vectors
Input f-structure HF-vectors
Slot f-structure representation (HF vectors)
(b)

Figure 2: Phrasal Network. **(a)**: the architecture of a phrasal network. An oval in the hidden layer may represent multiple hidden units. **(b)**: details of the lowest two layers. The unshaded slots represent the input f-structure. The shaded ones represent the context f-structure.

1 vector with exactly one activated element indicating the grammatical relation of the input structure.

3. Lexical vector of the head of the output f-structure's parent structure (*p-head*). Sometimes, one input f-structure may be responsible for the generation of multiple target f-structures at different levels. For example, [$_{sentence}$ GOODBYE] corresponds to both [$_{sentence}$ AUF [$_{obj}$ WIEDERHOEREN]] and its sub-structure [$_{obj}$ WIEDERHOEREN] in the training samples. This input serves as a stack pointer, indicating the level at which the output f-structure should be generated.

The HF-vectors at the input layer are the local representations for the words and features in an f-structure. The activation patterns of the slots at the **feature layer** can be viewed as the automatically learned distributed representation of the input HF-vectors (Miikkulainen & Dyer 89). The input slots have one-to-one connections to the feature slots [Figure 2(b)]. The slot-slot connections share weights in such a way that the connection from the *i*th unit in slot A at the input layer to the *j*th unit in slot A at the feature layer has the same strength as the connection from the *i*th unit in slot B at the input layer to the *j*th unit in slot B at the feature layer. The weight sharing guarantees that the same HF-vector at different input slots will result in the same pattern in their corresponding feature slots.

The **output layer** of a phrasal network has three parts:

1. the **HF-vector** of the f-structure to be generated. From this vector the head and the terminal features of the target f-structure can be recovered.
2. the **Sub-Structures' Input Specifiers**. It consists of slots of 0-1 vectors. Each slot has at most one element being activated. And each slot corresponds to a sub-structure of a specific role of the target f-structure. The role of the sub-structure is implied by the position of the slot in the output layer. Each vector of *sub-structures' input specifier* is of the size ($||$ *input layer slots* $||+1$). For an output slot in *sub-structures' input specifier*, if it has one activated element, then the sub-structure with the corresponding role should be included as a part of the desired output f-structure. The position of the activated element in the slot indicates the input sub-structure (as specified by the slot number in the input layer) that is the counterpart of (and therefore is responsible for the generation of) the target sub-structure, or *nil* when no input sub-structure is a counterpart of the output sub-structure. If a network does not activate any element in an output slot, then the slot's corresponding sub-structure should not be expected as a part of the desired target f-structure.
3. the **Sub-Structures' Categories**. It consists of slots of 0-1 vectors. There can be at most one element being activated in each slot, specifying one of the seven phrasal categories for the corresponding target sub-structure.

the HF-vector of the input f-structure with different roles always occupies the first slot.

According to a network's output, the controller can build sub-structures recursively by assigning subsequent sub-structure mapping tasks to the networks of the categories specified in the output of *sub-structures' categories* at the output layer. The input f-structures of those mapping tasks are specified in the output of *sub-structures' input specifiers*. By combining the recursively built sub-structures and the head and the terminal features from the output HF-vector, the desired target f-structure can be produced.

4 An Example

The example in [Figure 4] illustrates how the system works.

In step (0), the controller first activates the IP network with the *source* input f-structure. There is no *context input* for the IP network, since the sentential f-structures are the top level f-structures in our task. From the network's output, the controller knows that the head of the IP is NIL[7]. It also generates the sentential feature *(MOOD *DECLARATIVE)*. And it interprets the output as that the only sub-structure of the sentence is a German VP, whose English counterpart is the (non-proper) sub-structure with the head *WOULD*[8]. Therefore it builds the target f-structure framework [NIL (MOOD *DECLARATIVE) [$_{sentence}$ *]], and activates the VP network in step (1). Upon receiving the VP sub-structure returned from step (1), it combines that sub-structure with the f-structure framework, and collapses the NIL-headed f-structure to form the assembled f-structure shown as the output in step (0).

[4]The sub-structure's input specifier and category are combined into a tuple here.

[5]The number in the parenthesis indicates the subsequent step of network activation for this sub-structure.

[6]*P-head* is the head of the target f-structure's parent structure

[7]NIL-headed f-structure happens only when there is only one sub-structure or when there is an *xcomp* sub-structure. The NIL-headed f-structure must collapse into the only sub-structure in the first case, or into the *xcomp* sub-structure in the second case. All terminal features and other sub-structures are moved into the *collapsed-into* sub-structure during collapsing.

[8]The network actually specifies the slot at the input layer instead of the lexical item *WOULD*.

Source Sentence: *I would like to register for the conference*
Source F-structure: $[_{sentence}$ $[_{subj}$ I] WOULD $[_{xcomp}$ $[_{subj}$ I] LIKE $[_{xcomp}$ $[_{subj}$ I]
REGISTER $[_{pp-adj}$ FOR $[_{obj}$ $[_{det}$ THE] CONFERENCE]]]]]
Target Sentence: *Ich wuerd mich gerne zur Konferenz anmelde*

(0) IP network
Input:
source: $[_{sentence}$ $[_{subj}$ I] WOULD $[_{xcomp}$ $[_{subj}$ I] LIKE $[_{xcomp}$ REGISTER]]]
Output:
head: NIL
subs: sentence <WOULD VP>[4] (1)[5]
features: (MOOD *DECLARATIVE)
F-structure assembled by the controller:
$[_{sentence}$ $[_{subj}$ PRONOUN] WERDE
$[_{xcomp}$ $[_{subj}$ PRONOUN] $[_{adj}$ GERNE] ANMELDEN $[_{obj}$ PRONOUN]
$[_{pp-adj}$ FUER $[_{obj}$ $[_{det}$ DER] KONFERENZ]]]]

(1) VP network
Input:
source: $[_{sentence}$ $[_{subj}$ I] WOULD $[_{xcomp}$ $[_{subj}$ I] LIKE $[_{xcomp}$ REGISTER]]]
context: NIL
role: sentence
p-head:[6] NIL
Output:
head: WERDE

subs:	subj <I	NP>	(2)
	xcomp <LIKE	VP>	(3)

features: ((CAT V) (PERSON 1) (MODAL +) (FORM FIN) ...)
F-structure assembled by the controller:
[$[_{subj}$ PRONOUN] WERDE
[$[_{subj}$ PRONOUN] $[_{adj}$ GERNE] ANMELDEN $[_{obj}$ PRONOUN]
$[_{pp-adj}$ FUER $[_{obj}$ $[_{det}$ DER] KONFERENZ]]]]

(2) NP network
Input:
source: $[_{subj}$ I]
context: $[_{sentence}$ $[_{subj}$ I] WOULD $[_{xcomp}$ LIKE]]
role: Subject
p-head: WERDE
Output:
head: PRONOUN
subs: NIL
features: ((CAT N) (PERSON 1) (NUMBER SG) (CASE N))
F-structure assembled by the controller:
$[_{subj}$ PRONOUN]

......

Figure 3: An example of connectionist transfer.

In step (1), the input *source* was determined in step (0), since the *sentence* sub-structure's head was *"WOULD"* according to the IP network's *sub-structure's input specifier* in step (0). The *context* input is NIL because the *source* f-structure does not have a parent f-structure. The input *role* has the value *sentence* because the slot position of the output sub-structure in step (0) implies the grammatical relation of the sub-structure is *sentence*. The input *p-head* is NIL because the head of target f-structure in step (0) was NIL as specified by the output HF-vector there.

The VP network maps the input f-structure to its German counterpart by specifying (a) the head of the German VP structure *WERDE* and the terminal features of the German VP structure in the output HF-vector, and (b) the input specifiers and the categories of the sub-structures of the target German VP f-structure. To build detailed sub-structures for this VP f-structure, the controller will activate the NP network with the input of the English sub-structure with the head *"I"* and the VP network with the input English sub-structure with the head *"LIKE"* in step (2) and (3), and combine the sub-structures returned from step (2) and (3) into the f-structure framework $[[_{subj}$ *] WERDE $[_{xcomp}$ *]]. The combined structure is then returned to step (0) to be integrated into the top level f-structure framework there.

Step (2) works in the similar way to generate the NP subject f-structure $[_{subj}$ PRONOUN] as the sub-structure of the VP f-structure framework in step (1). We do not go into the details for this step here, and we omit the further steps in the process of generating the target f-structure.

5 Training, Testing and Performance

From the 300 sentential f-structure pairs, we extracted all the German NP sub-structures, their grammatical relations and their parent structures' heads. We labeled their English counterparts[9]. These were all the information required for the training of the NP network. About 700 samples for the NP networks were created this way. The training samples for the other networks were prepared in the same way. The NP network had the most samples, while the MP network had the least of 89 samples. Standard back-propagation was used to train the networks. We also tried the information-theoretical networks (Gorin *et al.* 91) to generate the head of a target structure in the HF-vector, which required less training time and achieved comparable performance as the network trained with pure back-propagation algorithm(Wang 94). The training took 500 to 2000 epochs for different networks, and the training time ranged from one hour to three days on DECStation 5000. The mapper achieved 92.4% accuracy on the training data[10].

Learnability: The connectionist f-structure mapping described above did not require any hand-crafted rules or representations. The structure transfer was learned automatically. By clustering the distributed representations of words learned by the networks, i.e., the activation patterns of a feature slot when a lexical item was presented to its connected input slot, we had some interesting findings about what was learned by the networks. One of them was that the feature patterns for English nouns in the DP network were clustered into three classes, which reflected the three genders of German nouns: the German translations of the words in each class were roughly of the same gender. Another finding was about the classification of verbs. When we clustered the feature patterns for verbs in the VP networks, we found some intransitive verbs like *register* in the same class as most of the transitive verbs. This seemly strange classification is not odd at all if we consider the fact that the German translation for *register*, "*anmelden*", is a transitive verb. These two independent findings reveal the networks' ability to discover some linguistic features of the target language and use it in the representation of an entity of the source language which does not possess those features. This is exactly what a symbolic transfer are supposed to do: using an intermediate representation which reflects the linguistic features of the two languages in question (even if one of the languages may have degenerated form for a specific feature,) and thus being able to make a "transfer" at both the lexical and structural level into corresponding structure in the target language. Our system learned the intermediate representation automatically, although it was not expressed explicitly in symbolic forms but encoded in the networks' activation patterns. Because the development of this representation was integrated into the process of automatic learning of f-structure mapping, it tended to include in the intermediate representation the important language specific linguistic features which were directly relevant for the ultimate purpose of structure transfer. In the other words, the learning of the intermediate representation was focused on the purpose of improving the transfer performance. This is one of the biggest advantage of this approach over the hand-crafted intermediate representation.

Scalability: We did a preliminary scalability experiment. We extended the source and tar-

[9]An NP's counterpart is not necessary to be an NP.

[10]A source language f-structure is said to be *accurately* mapped if the generated target language f-structure is exactly the same as desired in the sample. We understand that it is not appropriate to talk about *accuracy* of translation. There might be many different but correct translations. We report the accuracy here to asset the networks' performance on the mapping to the f-structure of a *specified* translation.

get language lexicon by 2%, and made 30 new f-structures with these new lexical items. Trying to scale up from what was already learned, we froze all but the input-feature connections, trained the network for about 40 epochs with the new data, then fine-tuned all the connections with old and new data for a few epochs. In doing so, we let the networks first learn the new words to derive their distributed representations, and then learn the structure mapping for the new data later. This approach was based on the observation that a big portion of the new English words were translated to some German words already in the lexicon, which in turn was translated from some English words in the old training data. These old English words were mostly the synonyms of the new English words. By freezing the other connections and training only the input-feature connections, we hoped the networks to be able to develop the distributed representation for a new word similar to the already-learned representations of its synonyms.

This approach greatly reduced the learning time for new words, since the one layer back-propagation was much fast than the full-blown learning. The mapper with the new phrasal networks that were retrained this way achieved 83.3% accuracy on the new data, without affecting the performance on the old data.

Generalizability: A separate set of data was used to test the generalization performance of the system. The testing data was collected for an independent connectionist parsing task (Jain 91) from people not associated with our researches. The data was compared with the training corpus, and the sentences that appeared in the training data were removed. An LR parser (Tomita 90) parsed the sentences to English f-structures. The English were translated into German manually, and the translations were parsed by a German LR-parser. We picked the most probable structure when a parsing result was ambiguous. There were 154 f-structure pairs after we eliminated the wrongly-parsed sentences. The mapper achieved 61.7% accuracy on the testing data. Considering the limited number of training samples, this performance was encouraging. Previous research as in (Chrisman 91) did not generalize to deal with unseen data.

6 Discussion

The application of the connectionist transfer described in this paper has its restrictions. First, it requires well-formed f-structures for both the input and output sentences. This greatly limits the applicable domain of the approach to well-structured "clean" languages. We have tried to apply the approach to our spoken language scheduling data (Suhm *et al.* 95) recently. Since the data is pervasive with ungrammatical utterances, noises, false starts, etc, it is almost impossible to automatically derive the well-formed f-structures from the utterances with a parser.

Another restriction is that this approached can only achieve satisfiable performance when the input and output languages are similar, in the sense that the translation equivalents in the two languages mostly have similar recursive f-structures. Although the system can deal with structurally different input/output sentences, like the aforementioned example of $[_{sentence}$ GOODBYE$]$ and $[_{sentence}$ AUF $[_{obj}$ WIEDERHOEREN$]]$, we believe that the performance would drop significantly if drastic structure differences[11] between translation equivalents are pervasive for the two languages in question. Fortunately, as shown by our data, the structural difference between English and German is not so drastic to ruin our system's performance.

Although we had done some scalability experiment, it is unclear how the system will perform if we increase the lexicon significantly instead of by 2%. Because of the limitation of available data, we found it very difficult to conduct scalability experiments with much more expanded lexicon. We hope that with stable incremental performance, the system can be gradually and easily retrained to deal with more complicated problems.

We did not address the problem of ambiguity in this research. The networks were trained to map from the most probable f-structures of English sentences to the most probable f-structures of their German counterparts. Therefore it is necessary to have a mechanism to dissolve the ambiguities and feed the most probable f-structures of input sentences to the mapper. This was done by hand-picking the most probable English f-structures in our research.

7 Conclusion

Aiming at the difficulties in symbolic transfer, we have proposed a connectionist transfer system that maps between f-structures of two languages. It can discover meaningful linguistic features by learning. Its performance is promising with respect to learnability, scalability and generalizability.

References

(Bresnan 82) J. Bresnan. *The Mental Representation of Grammatical Relations.* MIT Press, Cambridge, Massachusetts, 1982.

(Chrisman 91) Lonnie Chrisman. Learning recursive distributed representations for holistic computation. *Connection Science*, 3(4):345–366, 1991.

(Gorin *et al.* 91) A. L. Gorin, S. E. Levinson, and A. E. Goldman. Adptive acquisition of

[11] Such as f-structures with heads that are not translation equivalents and/or have very different subcategorization frames.

language. *Computer Speech and Language*, (5):101–132, 1991.

(Jain 91) A. N. Jain. Parsec: A connectionist learning architecture for parsing spoken language. Technical Report CMU-CS-91-208, Carnegie Mellon University, 1991.

(Miikkulainen & Dyer 89) R. Miikkulainen and M. G. Dyer. A modular neural network architecture for sequential paraphrasing of script-based stories. In *Proceedings of the International Joint Conference on Neural Networks*. IEEE, 1989.

(Osterholtz & etal. 92) L. Osterholtz and et al. Janus: a multi-lingual speech to speech translation system. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 209–212. IEEE, 1992.

(S. Nirenbrug & Tuker 87) V. Raskin S. Nirenbrug and A. B. Tuker. The structure of interlingua in translator. In S. Nirenberg, editor, *Machine Translation: Theoretical and Methodological Issues*. Cambridge University Press, 1987.

(Suhm *et al.* 95) B. Suhm, P.Geutner, T. Kemp, A. Lavie, L. Mayfield, A. McNair, I. Rogina, T. Schultz, T. Sloboda, W. Ward, M. Woszczyna, and A. Waibel. JANUS: Towards multilingual spoken language translation. In *Proceedings of the ARPA Speech Spoken Language Technology Workshop, Austin, TX, 1995*, 1995.

(Tomita 90) M. Tomita. The generalized lr parser/compiler. In *proceedings of the 13th International Conference on Computational Linguistics*, Helsinki, 1990.

(Waibel & etal. 91) A. Waibel and et al. Janus: A speech-to-speech translation system using connectionist and symbolic processing strategies. In *IEEE International Conference on Acoustics Speech and Signal Processing*. IEEE, May 1991.

(Wang 94) Ye-Yi Wang. Dual-coding theory and connectionist lexical selection. In *Proceedings of the 32nd Annual Meeting of the ACL, student session*, pages 325–327, 1994.

(White 87) J. S. White. The research environment in the metal project. In S. Nirenberg, editor, *Machine Translation: Theoretical and Methodological Issues*. Cambridge University Press, 1987.

(Witkam 83) A.P.M. Witkam. *Distributed language translation: feasibility study of a multilingual facility for vadiotex information networks*. BSO, Utrecht, 1983.