## Rapid Prototyping of Robust Language Understanding Modules for Spoken Dialogue Systems


†Yuichiro Fukubayashi, †Kazunori Komatani, ‡Mikio Nakano, ‡Kotaro Funakoshi, ‡Hiroshi Tsujino, †Tetsuya Ogata, †Hiroshi G. Okuno

†Graduate School of Informatics, Kyoto University
‡Honda Research Institute Japan Co., Ltd.


# Background

- In an early phase of the development of spoken dialogue systems...
  - Large amounts of data are required for robust language understanding (LU).
  - However, large amounts of data are not available.
  - **To construct robust LU modules needs a lot of efforts and is time-consuming.**



# Goal

**Rapid prototyping of LU modules**

1. Robust against various expressions.
2. Easy to construct (requires less training data).

**More robust LU modules with less training data.**



# Related Work

**Rule- or grammar-based approach**

- keyword spotting (e.g. VoiceXML)
- heuristic rules (Seneff, 1992)

☹ **Less robust against various expressions.**
- Cannot reject automatic speech recognition (ASR) errors.
- Keyword spotting does not consider grammatical rules.

☺ **Easy to construct (requires less data)**
- Preparing grammars takes less efforts.



# Related Work

**Stochastic approach**

- corpus-based (Sudoh, 2005; He, 2005)
- Weighted Finite State Transducer (WFST)-based (Potamianos, 2004; Wutiwiwatchai, 2004)

☺ **Robust against various expressions.**
- Reject ASR errors with trained LU modules.
- WFST is considering grammatical rules.

☹ **Not easy to construct (requires much data)**
- Large amount of data for training is required for robust LU.
- Collecting a large amount of data takes much effort.



# Our Approach

**WFST-based LU with simpler weightings**

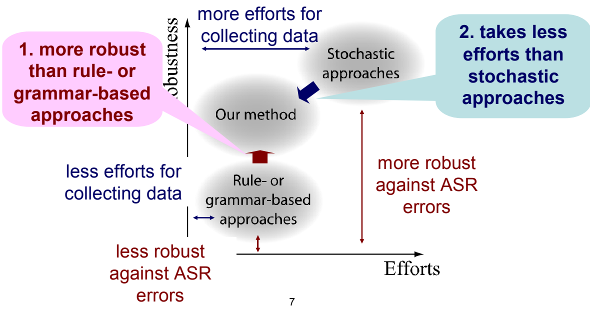
- Weighting should be simpler than conventional methods.
- Optimal parameters are obtained with small amount of data.

☺ **Robust against various expressions.**
- Reject ASR errors with trained WFST.
- WFST is considering grammatical rules.

☺ **Easy to construct (requires less data)**
- Preparing grammars takes less efforts.
- Required data for training is small.

# Position of Our Method

- A modest and realistic approach.

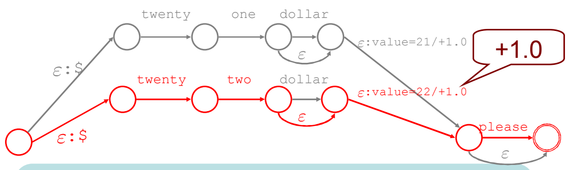





# WFST-based Approach

# WFST-based Language Understanding

- WFST accepts ASR outputs as its input.

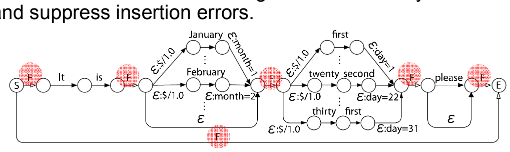



Input: twenty two, please
Output: $ twenty two value=22 please
Cumulative weight: +1.0
LU result: value=22



# FILLER Transition

- FILLER transition accepts any words.
  - FILLER transition enables to ignore unnecessary words for LU and suppress insertion errors.




* **F** represents 0 or more FILLER transitions.

Input = "It is February twenty second, please"

| LU outputs | | | | | |
|---|---|---|---|---|---|
| It | is | February | twenty | second | please |
| It | is | *FILLER* | twenty | second | please |
| It | is | *FILLER* | twenty | second | *FILLER* |
| *FILLER* | *FILLER* | *FILLER* | *FILLER* | *FILLER* | *FILLER* |



# Issue: Design of Weighting Schemes

- The path with highest cumulative weight is selected from various output sequences.

| LU output | | | | | | LU result | *w* |
|---|---|---|---|---|---|---|---|
| It | is | February | twenty | second | please | month=2, day=22 | 2.0 |
| It | is | *FILLER* | twenty | second | please | day=22 | 1.0 |
| It | is | *FILLER* | twenty | second | *FILLER* | day=22 | 1.0 |
| *FILLER* | *FILLER* | *FILLER* | *FILLER* | *FILLER* | *FILLER* | - | 1.0 |

**Requirements for weighting schemes**

1. Robust against various expressions (ASR errors).
2. Simple features for weighting.

Reduce the amount of data for training.



# Weighting Schemes for WFST

## Outline of Our Method

ASR N-best output

| | |
|---|---|
| 1 | It is … |
| 2 | Ah it is … |
| 3 | It is … |
| … | … |
| N | … |

WFST with optimal parameters

WFST output with cumulative weight

| | cumulative weight | WFST output |
|---|---|---|
| 1 | $w^1$ | … |
| 2 | $w^2$ | … |
| 3 | $w^3$ | … |
| … | … | … |
| N | $w^N$ | … |

argmax $w^i$

Language Understanding Result

training

Training data

Minimize concept error rate (CER) by changing parameters

Optimal parameters

- weighting schemes
- coefficients

13

## Weighting on Two Levels

1. Weighting for ASR outputs
2. Weighting for concepts

Weighting for ASR N-best outputs

Weighting for accepted words

Weighting for concepts

$$w^i = w_s^i + \alpha_w \sum w_w + \alpha_c \sum w_c$$

$w^i$ is the cumulative weight of $i$-th sentence of ASR N-best.

14

## Parameters for Training

- Five kinds of parameters to determine.

**ASR N-best** N=1 or 10?

**Accepted words** Which weighting scheme to use?

**Concept** Which weighting scheme to use?

$$w^i = w_s^i + \alpha_w \sum w_w + \alpha_c \sum w_c$$

**Coefficient** $\alpha_w$=0 or 1.0?

**Coefficient** $\alpha_c$=0 or 1.0?

15

## 1. Weighing for ASR outputs
## Weighting for ASR N-Best Outputs

$$w^i = w_s^i + \alpha_w \sum w_w + \alpha_c \sum w_c$$

Reflects the reliability of the ASR output.

$$w_s^i = \frac{e^{\beta \cdot score_i}}{\sum_j e^{\beta \cdot score_j}}$$

$\beta$ is a coefficient for smoothing, $score_i$ is the log-scaled score of $i$-th ASR output.

| $i$ | $w_s^i$ |
|---|---|
| 1 | 0.78 |
| 2 | 0.10 |
| 3 | 0.09 |
| … | … |

More reliable output is assigned higher weight.

16

## 1. Weighting for ASR outputs
## Weighting for Accepted Words

$$w^i = w_s^i + \alpha_w \sum w_w + \alpha_c \sum w_c$$

twenty/$w_w$

Three candidates of weighting schemes.

**1.word (const.):** $w_w=1.0$

**2.word (#phone):** $w_w=l(W)$

The normalized length of phonemes in $W$ $(0<l(W)\leq 1)$.

- Sequences with more words are simply preferred.

**3.word (CM):** $w_w=CM(W)-\theta_w$

Confidence measure for $W$ $(0<CM(W)\leq 1)$.

$\theta_w$ is the threshold for determining whether $W$ is accepted or not.

- Longer and reliable sequences are preferred.

17

## 2. Weighting for Concepts

$$w^i = w_s^i + \alpha_w \sum w_w + \alpha_c \sum w_c$$

Three candidates of weighting schemes.

**1.cpt(const.):** $w_c=1.0$

value=22/$w_c$

- A preference for sequences with more concepts.

**2.cpt(avg):**

$w_c=(\Sigma_W CM(W)-\theta_c)/\#W$

**3.cpt(#pCM(avg)):**

$w_c=\Sigma_W (CM(W) l(W)-\theta_c)/\#W$

$W$: set of words in the concept
$\#W$: the number of words in $W$

- A preference for longer and reliable concepts.

18

## Example: Weighting Scheme for Accepted Words

**Candidates for accepted words**

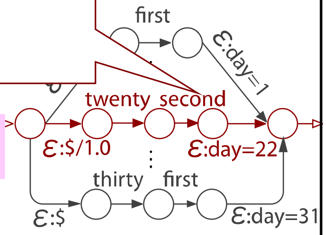
**const.** : $w_w$=1.0

**#phone** : $w_w$=0.5

**CM** : $w_w$=0.9-$\theta_w$

length of word: $l(\text{"second"})$

CM of word: $CM(\text{"second"})$






## Example: Weighting Scheme for Concepts

**average of CM**

$$\frac{CM(\text{"twenty"})+CM(\text{"second"})}{2}$$

**average of length*CM**

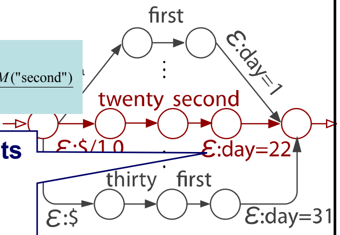
$$\frac{l(\text{"twenty"})CM(\text{"twenty"})+l(\text{"second"})CM(\text{"second"})}{2}$$

**Candidates for concepts**

**const.** : $w_c$=1.0

**avg** : $w_c$=0.95-$\theta_c$

**#pCM(avg)** : $w_c$=0.525-$\theta_c$






## Training: Determine Parameters

Training data → Minimize concept error rate (CER) by changing parameters → Determine optimal parameter sets

| $N$ | $\alpha_w$ | word | $\alpha_c$ | cpt | CER |
|---|---|---|---|---|---|
| **10** | **1.0** | **CM-0.6** | **1.0** | **#pCM(avg)-0.4** | **29.2** |
| 1 | 1.0 | const. | 1.0 | avg | 32.6 |
| 10 | 1.0 | CM-0.4 | 0 | - | 36.8 |
| … | … | … | … | … | … |

Optimal parameters

ASR N-best N=10

Accepted words word(CM)-0.6

Concept #pCM(avg)-0.4

$$w^i = w_s^i + \alpha_w \sum w_w + \alpha_c \sum w_c$$

Coefficient $\alpha_w$=1.0

Coefficient $\alpha_c$=1.0



## Cumulative Weight

ASR N-best N=10

Accepted words word(CM)-0.6

Concept cpt(#pCM(avg))-0.4

Cumulative weight $w^i = w_s^i + 1.0(4.1-5\cdot 0.6) + 1.0(1.335-2\cdot 0.4)$

Coefficient $\alpha_w$=1.0

Coefficient $\alpha_c$=1.0

| | | | | | | |
|---|---|---|---|---|---|---|
| ASR output | No, | it | is | February | twenty | second |
| WFST output | *FILLER* | it | is | February | twenty | second |
| Concept | - | - | | month=2 | day=22 | |
| $CM(W)$ | 0.3 | 0.7 | 0.6 | 0.9 | 1.0 | 0.9 |
| $l(W)$ | 0.3 | 0.2 | 0.2 | 0.9 | 0.6 | 0.5 |
| $w_w$ | - | 0.7-0.6 | 0.6-0.6 | 0.9-0.6 | 1.0-0.6 | 0.9-0.6 |
| $w_c$ | - | - | | 0.81-0.4 | 0.525-0.4 | |



# Experiments and Evaluation

## Experimental Conditions

**Two different domains**

| | Video | Rent-a-car |
|---|---|---|
| Vocabulary size | 209 | 891 |
| Example sentences | 10000 | 40000 |
| # utterance | 4186 (25 x 8sessions) | 3364 (23 x 8sessions) |
| ASR Acc. | 83.9% | 65.7% |

- Rent-a-car is more complicated domain.
  - Larger vocabulary size
  - Lower ASR accuracy

# Experimental Conditions

- We evaluated the results with 4-fold cross validation.
  - Compared concept error rate (CER).
- Two baseline methods: simple keyword spotting
  1. **Grammar & spotting:** Grammar-based ASR + keyword spotting
  2. **SLM & spotting:** Statistical language model-based ASR + keyword spotting
  - Takes as many concepts as possible without considering grammatical rules.
  - Assuming a condition that a large amount of data is not available.

Example of keyword spotting in rent-a-car domain

| | | | | | | |
|---|---|---|---|---|---|---|
| ASR Output | From | June | third | uhm | FIT | please |

➡ month=6, day=3, car=FIT ('FIT' is the name of a car)



# Result 1
# Obtained Optimal Parameters

- The optimal parameters depend on the domain.
  - Complexity of domains reflects the parameters.

| Domain | $N$ | $\alpha_w$ | word | $\alpha_c$ | cpt |
|---|---|---|---|---|---|
| Video | 1 | 1.0 | **const.** | 0 | n/a |
| Rent-a-car | 10 | 1.0 | **CM-0.0** | 1.0 | **#pCM(avg)-0.8** |

- Recognition results are not reliable in rent-a-car domain.
  - Rent-a-car domain uses ASR 10-best output.
  - Rent-a-car domain uses confidence measure for weightings.



# Result 2
# Performance of WFST-based LU

**Lower CER with our method**

- Better performance with "SLM & spotting" than "Grammar & spotting" because of robust ASR.
- Further improvement with optimal weightings for WFST.

| Domain | Grammar & spotting | SLM & spotting | Our method |
|---|---|---|---|
| Video | 22.1 | 16.9 | 13.5 |
| Rent-a-car | 51.1 | 28.9 | 22.0 |

**due to SLM-based ASR** — **due to optimal weightings for WFST**

➡ **Our method outperformed two kinds of baseline.**

➡ **More robust than keyword spotting**

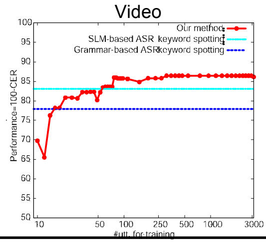


# Result 3
# Performance and Training Data

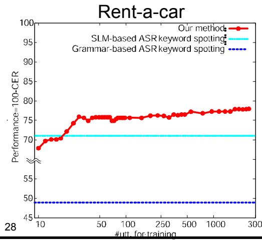
Our method outperformed baseline methods with **about 100 utterances.**

➡ **Easier to construct than stochastic methods.**
Conventional methods require several thousands of utterances.

Video




Rent-a-car






# Conclusion

**Rapidly prototyping robust LU modules.**

- WFST-based LU with simpler weighting.
- More robust than rule- or grammar-based methods.
- Easier to construct than stochastic methods.

**Experiments and Evaluation**

- Our method outperformed baseline methods with optimal weightings for WFST.
- Our method outperformed baseline methods with less utterances.
  - Conventional methods required several thousands of utterances.



# Future Work

- When to switch to stochastic approaches?
  - Stochastic approaches are more robust than our method if using large amounts of data.
  - How many data are needed for stochastic approach?