DSTA Multilingual Class 2021

Automatic Speech Recognition

Carnegie Mellon University
Language Technologies Institute

Table of Contents

- Speech recognition demo & Evaluation metrics
- (a bit) mathematical formulation of speech recognition
- Standard speech recognition pipeline

Demonstration (Japanese, Google voice search)

Sometimes working, sometimes failed

Reference)

I want to go to the CMU campus

Recognition result)

I want to go to the gym you can

Sometimes working, sometimes failed

Reference)

CMU大学のキャンパスに行きたいです

Recognition result)

CMU洋楽のキャンパスに行きたいです

Evaluation metric 1

- Sentence error rate
  - An entire sentence (utterance) is correct or not (100% error in the case below)

```
Reference)
I want to go to the CMU campus
Recognition result)
I want to go to the gym you can
```

  - Too strict, and needs to consider some local correctness

Evaluation metric 2

- Word error rate (WER)
  - Using edit distance word-by-word:

```
Reference)
I want to go to the CMU campus
Recognition result)
I want to go to the gym you can
```

- # insertion errors = 1, # substitution errors = 2, # of deletion errors = 0 ➡ Edit distance = 3
- Word error rate (%): Edit distance (=3) / # reference words (=8) * 100 = 37.5%
- How to compute WERs for languages that do not have word boundaries?
  - Chunking or using character error rate

Evaluation metric 3

- Character error rate (CER)
  - Using edit distance character-by-character:

    Reference)
    CMU大学のキャンパスに行きたいです
    Recognition result)
    CMU洋楽のキャンパスに行きたいです

    - # insertion errors = 0, # substitution errors = 2, # of deletion errors = 0
      ➡ Edit distance = 2
    - Character error rate (%): Edit distance (=2) / # reference words (=18) * 100 = 11.1%

Evaluation metric

- Other metrics
  - Phoneme error rate (need a pronunciation dictionary)
  - Frame error rate (need an alignment)

- NIST Speech Recognition Scoring Toolkit (SCTK)
  - ftp://jaguar.ncsl.nist.gov/pub/sctk-2.4.10-20151007-1312Z.tar.bz2

- WER can be > 100%, insertion case, deletion case

Speech recognition research is easy (?)

- We have WER or CER
- Objective, easy to obtain, very application-specific, single objective
- This can show the clear progress of technologies
- This would be one reason that the effectiveness of deep learning was first shown in speech

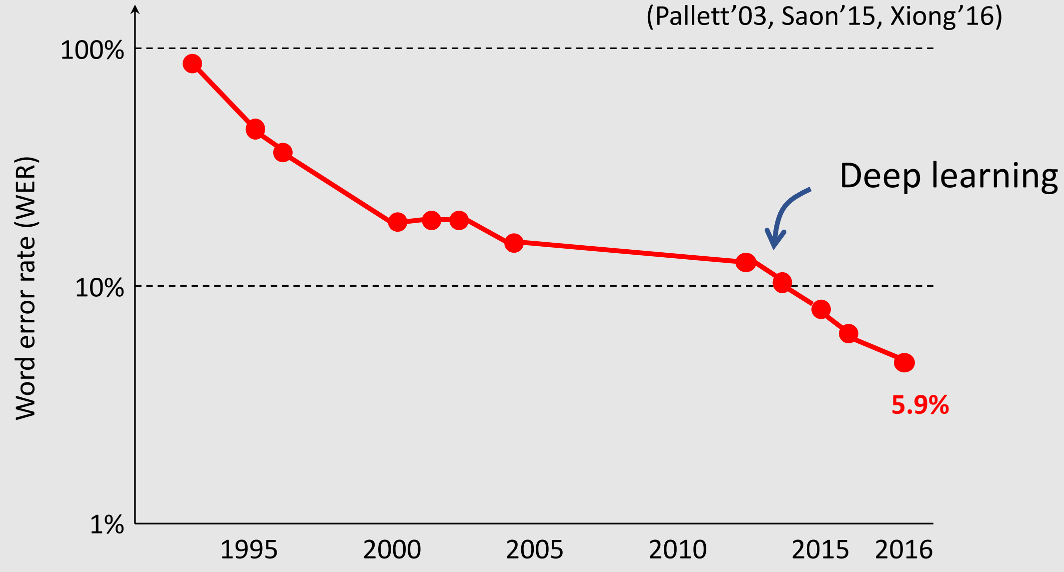
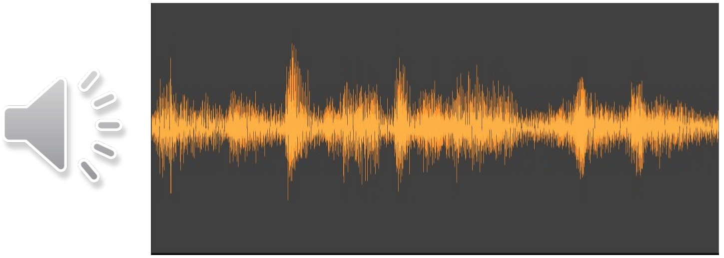


Table of Contents

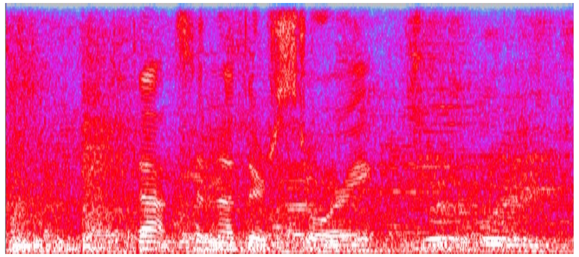
- Speech recognition demo & Evaluation metrics
- Standard speech recognition pipeline
- (a bit) mathematical formulation of speech recognition

Speech Recognition

- Mapping ***physical signal sequence*** to ***linguistic symbol sequence***



$$X = \{x_l \in \mathbb{Z} | l = 1, \ldots, L\}$$
$$L = 43263$$

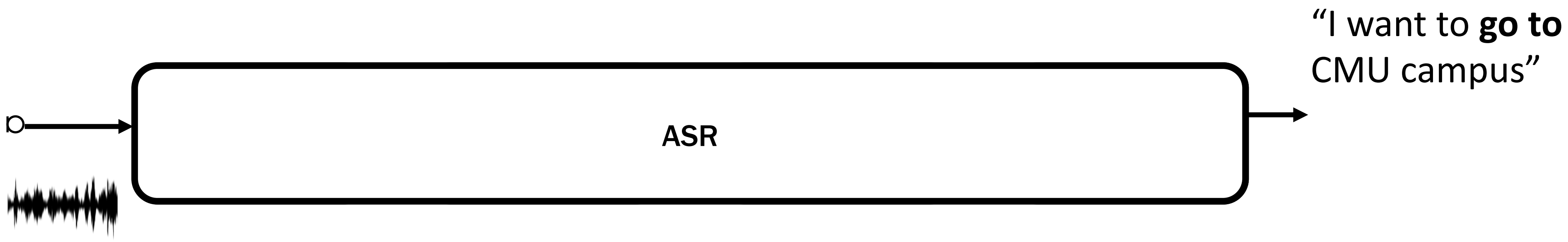


$$X = \{\mathbf{x}_t \in \mathbb{C}^D | t = 1, \ldots, T\}$$
$$T = 268$$

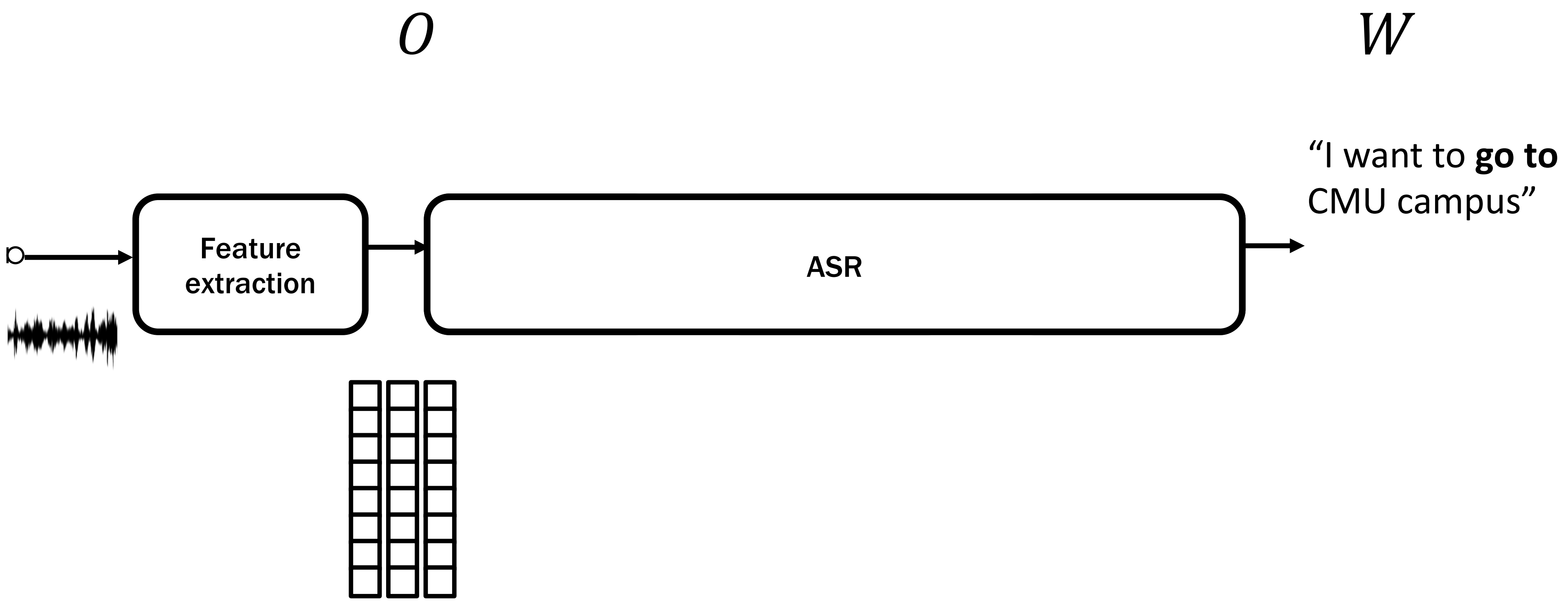
"That's another story"

$$W = \{w_n \in \mathcal{V} | n = 1, \ldots, N\}$$
$$N = 3$$

Automatic speech recognition



Automatic speech recognition

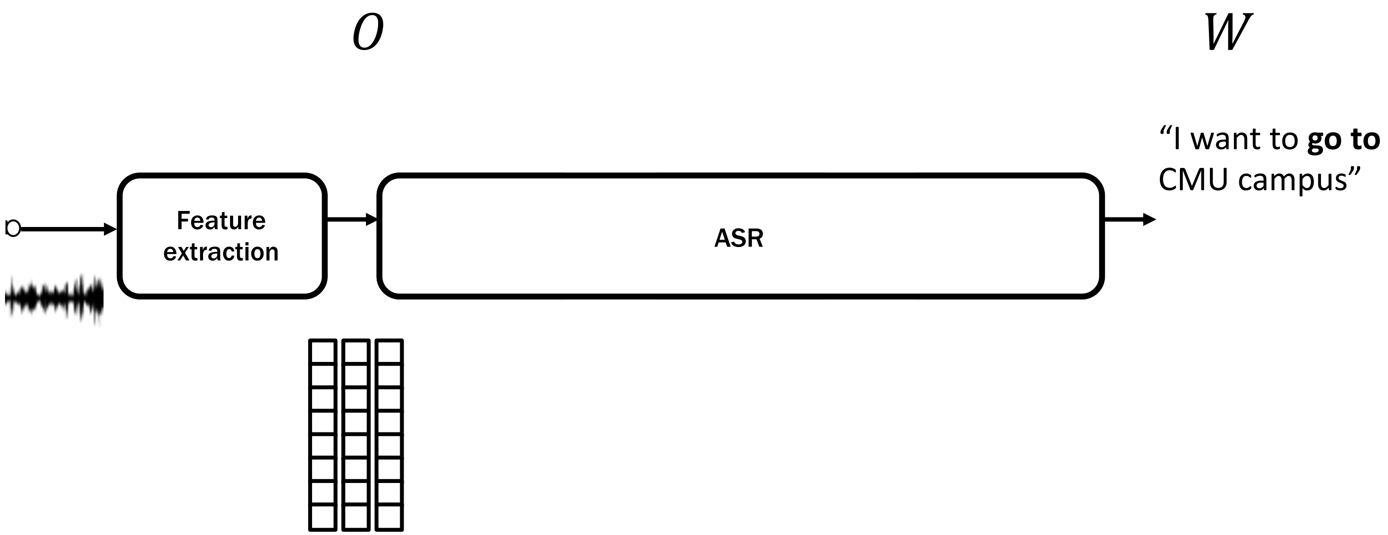


- Instead of starting from the waveform, we will often start from **speech features** (MFCC, etc.) through the **feature extraction** module
- Let’s think of the conversion from speech feature O to text W

Speech recognition with a probabilistic formulation

- **MAP decision theory**: Estimate the most probable word sequence $\hat{W}$ among all possible word sequences $\mathcal{W}$ (I'll omit the domain sometimes)

$$\hat{W} = \underset{W \in \mathcal{W}}{\operatorname{argmax}}\, p(W|O)$$



Probabilistic rules

- **Product rule**

$$p(x|y)p(y) = p(x,y)$$

- **Sum rule**

$$p(y) = \sum_x p(x,y)$$

- **Conditional independence assumption**

$$p(x|y,z) = p(x|z) \qquad p(x,y|z) = p(x|z)p(y|z)$$

How to obtain the posterior $p(W|O)$

- Noisy channel model
  - Regarding O as a probabilistic variable (noisy observation)
  - Use the product rule

$$\operatorname*{argmax}_W p(W|O) = \operatorname*{argmax}_W \frac{p(O|W)p(W)}{p(O)}$$

$$= \operatorname*{argmax}_W p(O|W)p(W)$$

Likelihood Prior

How to obtain the posterior $p(W|O)$

- Noisy channel model

$$\underset{W}{\operatorname{argmax}}\, p(W|O) = \underset{W}{\operatorname{argmax}} \frac{p(O|W)p(W)}{p(O)}$$
$$= \underset{W}{\operatorname{argmax}}\, p(O|W)p(W)$$

- Solving generating process of noisy observations!!
- Still difficult to deal with them….

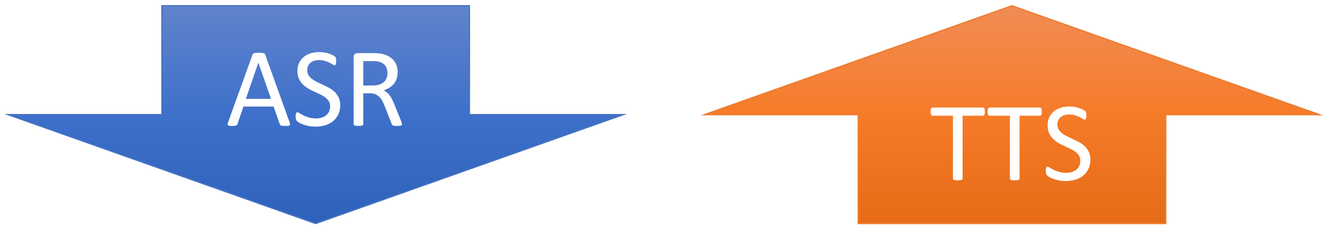
Speech <-> Text

Speech O:

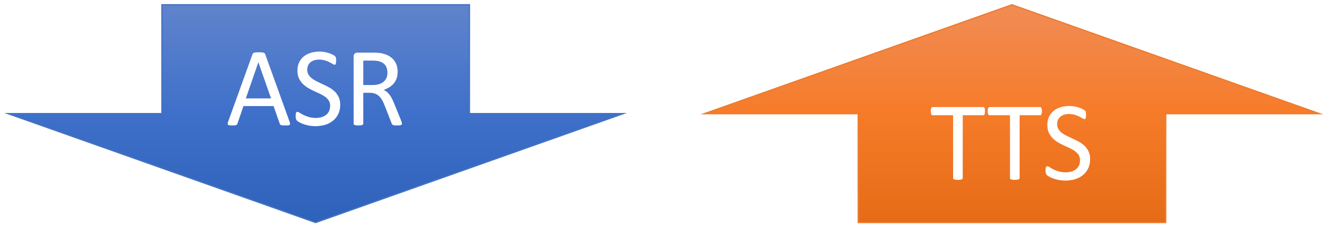
Text W: I want to go to the CMU campus

Speech <-> Phoneme <-> Text

Speech O:



Phoneme L: AY W AA N T T UW G OW T UW DH AH S IY EH M Y UW K AE M P AH S



Text W: I want to go to the CMU campus

How to obtain the posterior $p(W|O)$

- Further factorize the model with **phoneme**
  - Let $L = (l_i \in \{/\mathrm{AA}/, /\mathrm{AE}/, \cdots\} | i = 1, \cdots, J)$ be a phoneme sequence

$$\arg\max_{W} p(W|O)$$

How to obtain the posterior $p(W|O)$

- Further factorize the model with **phoneme**
  - Let $L = (l_i \in \{/\mathrm{AA}/, /\mathrm{AE}/, \cdots\} | i = 1, \cdots, J)$ be a phoneme sequence

$$\arg\max_W p(W|O) = \arg\max_W \sum_L p(W, L|O)$$

Sum rule

How to obtain the posterior $p(W|O)$

- Further factorize the model with **phoneme**
  - Let $L = (l_i \in \{/\mathrm{AA}/, /\mathrm{AE}/, \cdots\} | i = 1, \cdots, J)$ be a phoneme sequence

$$\arg\max_W p(W|O) = \arg\max_W \sum_L p(W, L|O)$$

Sum rule

$$= \arg\max_W \sum_L \frac{p(O|W, L)p(L|W)p(W)}{p(O)}$$

Product rule

How to obtain the posterior $p(W|O)$

- Further factorize the model with **phoneme**
  - Let $L = (l_i \in \{/\mathrm{AA}/, /\mathrm{AE}/, \cdots\} | i = 1, \cdots, J)$ be a phoneme sequence

$$\arg\max_W p(W|O) = \arg\max_W \sum_L p(W, L|O)$$

Sum rule

$$= \arg\max_W \sum_L \frac{p(O|W, L)p(L|W)p(W)}{p(O)}$$

Product rule

$$= \arg\max_W \sum_L p(O|W, L)p(L|W)p(W)$$

Ignore $p(O)$ as it does not depend on W

How to obtain the posterior $p(W|O)$

- Further factorize the model with **phoneme**
  - Let $L = (l_i \in \{/\mathrm{AA}/, /\mathrm{AE}/, \cdots\} | i = 1, \cdots, J)$ be a phoneme sequence

$$\arg\max_W p(W|O) = \arg\max_W \sum_L p(W, L|O)$$

Sum rule

$$= \arg\max_W \sum_L \frac{p(O|W,L)p(L|W)p(W)}{p(O)}$$

Product rule

$$= \arg\max_W \sum_L p(O|W,L)p(L|W)p(W)$$

Ignore $p(O)$ as it does not depend on W

$$= \arg\max_W \sum_L p(O|L)p(L|W)p(W)$$

Conditional independence assumption

Noisy channel model

$$\arg\max_{\mathrm{W}} p(W|O) = \arg\max_{W} p(O|W)p(W)$$
$$\approx \arg\max_{W} \sum_{L} p(O|L)p(L|W)p(W)$$

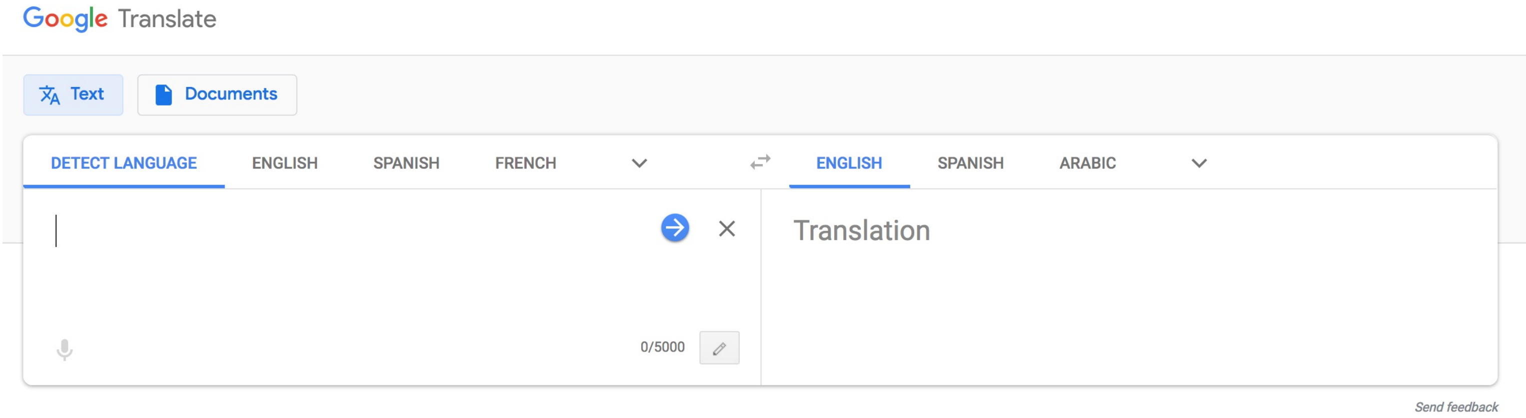
- **Speech recognition**
  - $p(O|L)$: Acoustic model (Hidden Markov model)
  - $p(L|W)$: Lexicon
  - $p(W)$: Language model (n-gram)

Noisy channel model

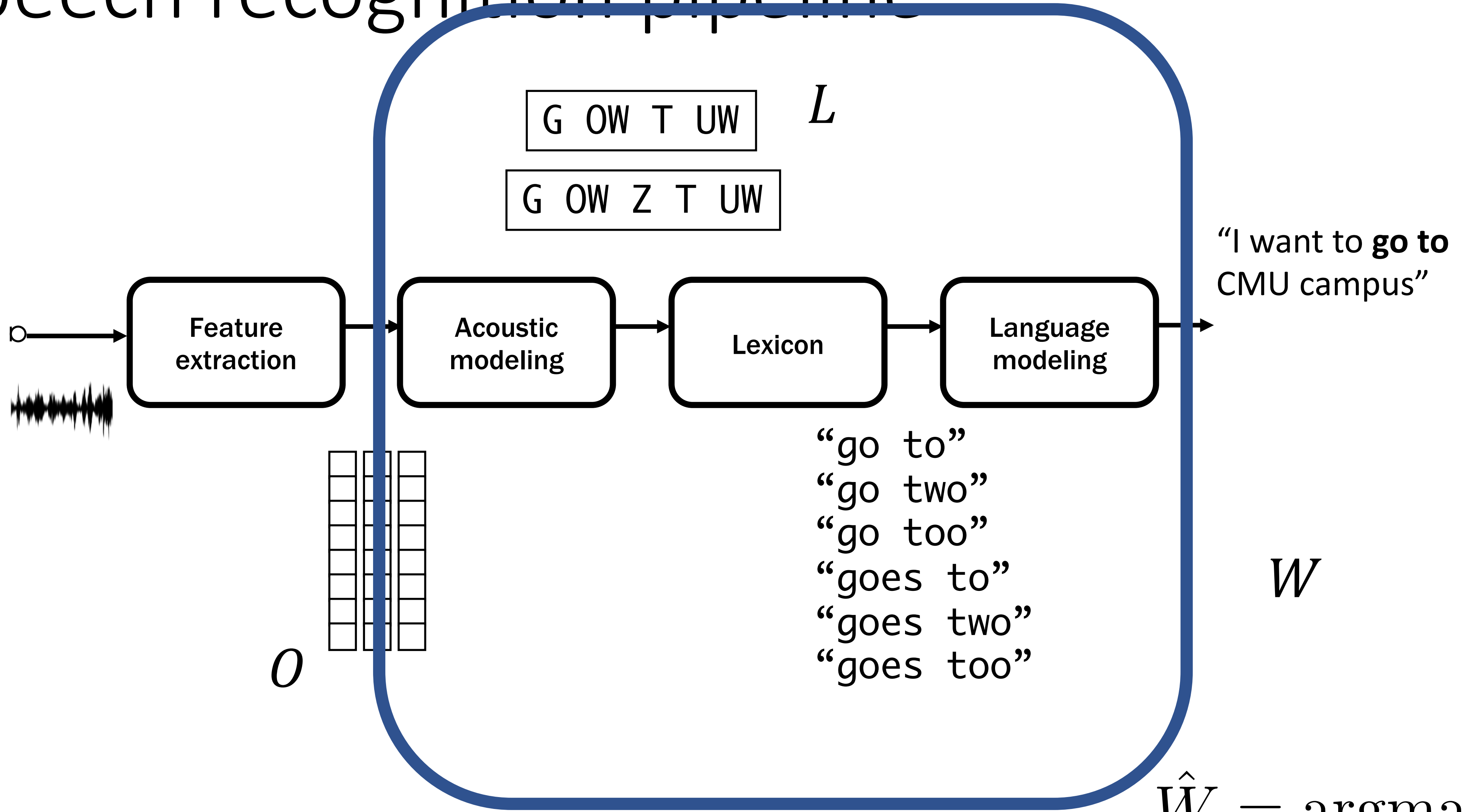
W: Target language text
Y: Source language text

$$\arg\max_{W} p(W|Y) = \arg\max_{W} p(Y|W)p(W)$$

- **Machine translation**
  - $p(Y|W)$: Translation model
  - $p(W)$: Language model



Speech recognition pipeline



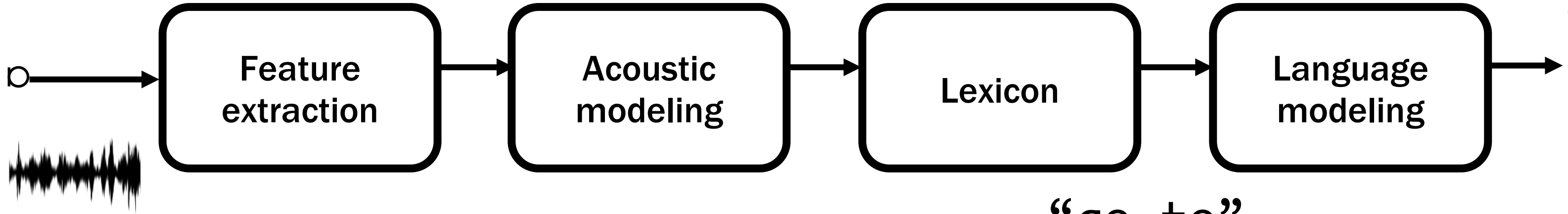
$$\hat{W} = \operatorname*{argmax}_{W \in \mathcal{W}} p(W|O)$$

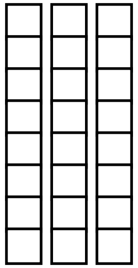
Speech recognition pipeline

G OW T UW

G OW Z T UW

"I want to **go to** CMU campus"





"go to"
"go two"
"go too"
"goes to"
"goes two"
"goes too"

$$\underset{W}{\operatorname{argmax}} \sum_{L} p(O|L)p(L|W)p(W)$$

$$p(O|L) \quad p(L|W) \quad p(W)$$

Please remember the noisy channel model

- **Factorization**
- **Conditional independence (Markov) assumptions**

We can elegantly factorize the speech recognition problem with a reasonable subproblem

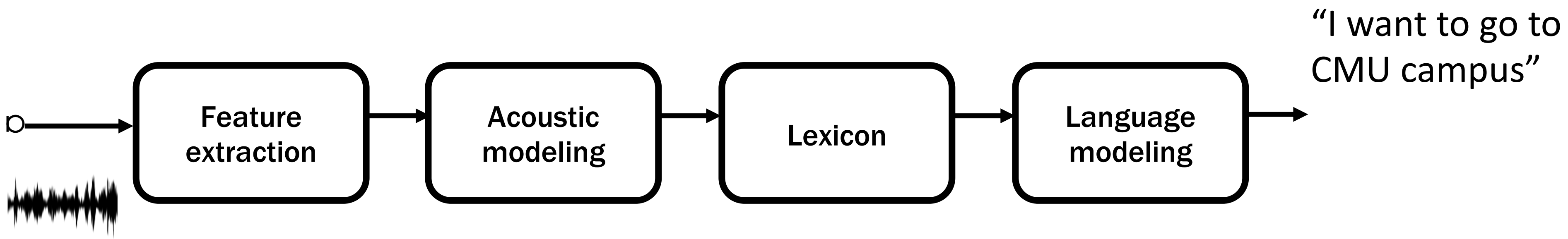
In the rest of courses, I'll introduce

- A bit further details of acoustic, lexicon, and language models
  - More and more factorization, conditional independence assumptions
- I'll skip these parts. If you want to know more about details, please check some other materials or take 11-751 "Speech Recognition and Understanding"
- For example, In 11-751 "Speech Recognition and Understanding", I'll spend 30% of the entire semester for this problem

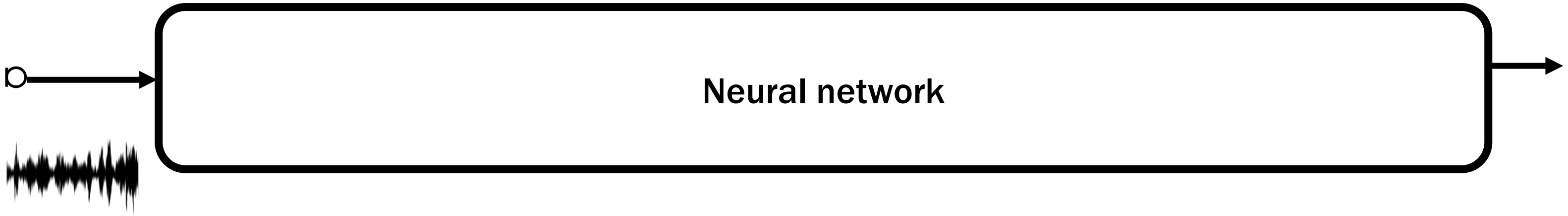
Table of Contents

- Speech recognition demo & Evaluation metrics
- (a bit) mathematical formulation of speech recognition
- Standard speech recognition pipeline

Main blocks

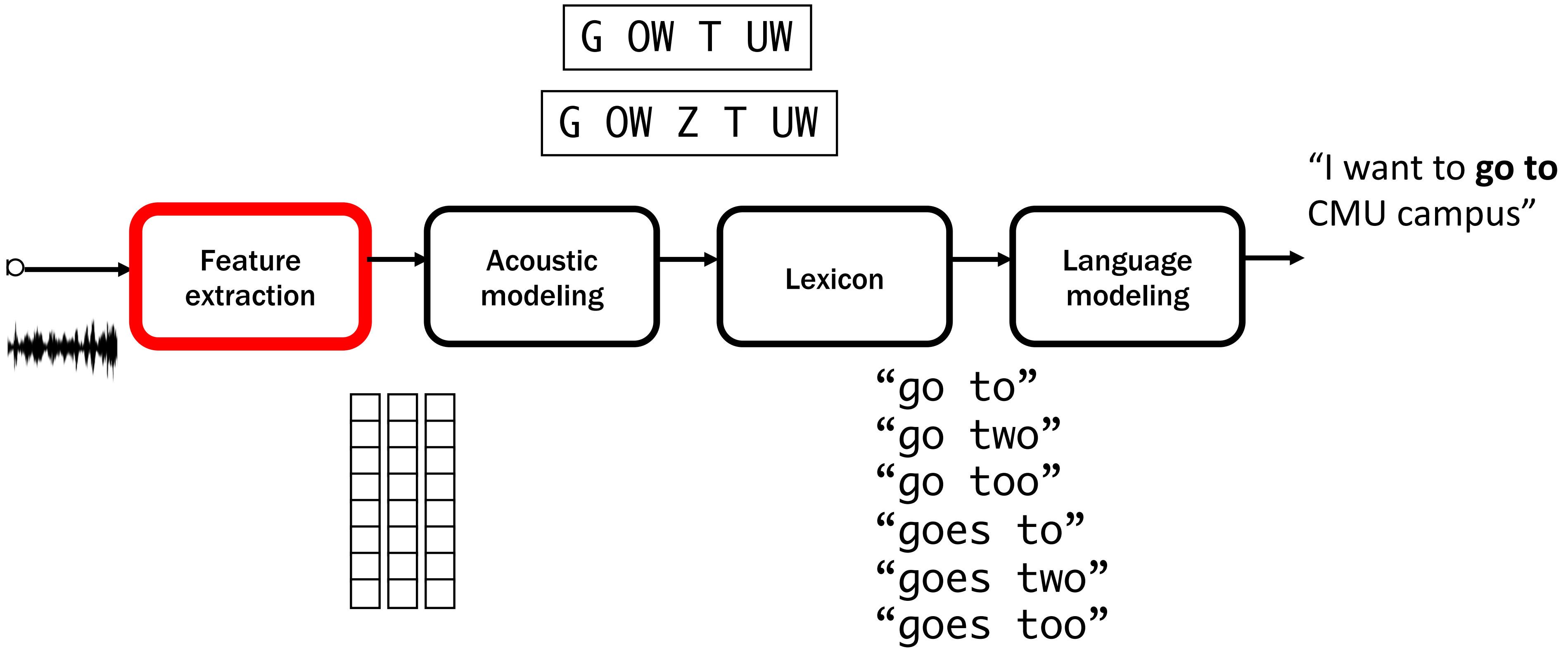

Feature extraction
Acoustic modeling
Lexicon
Language modeling
"I want to go to CMU campus"

Main blocks (end-to-end ASR)



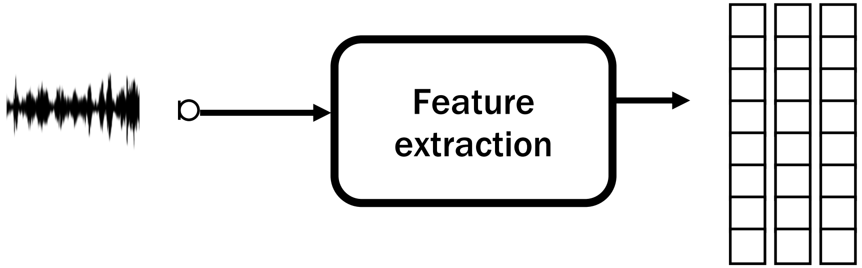
“I want to go to CMU campus”

Speech recognition pipeline



$p(O|L)$ $p(L|W)$ $p(W)$

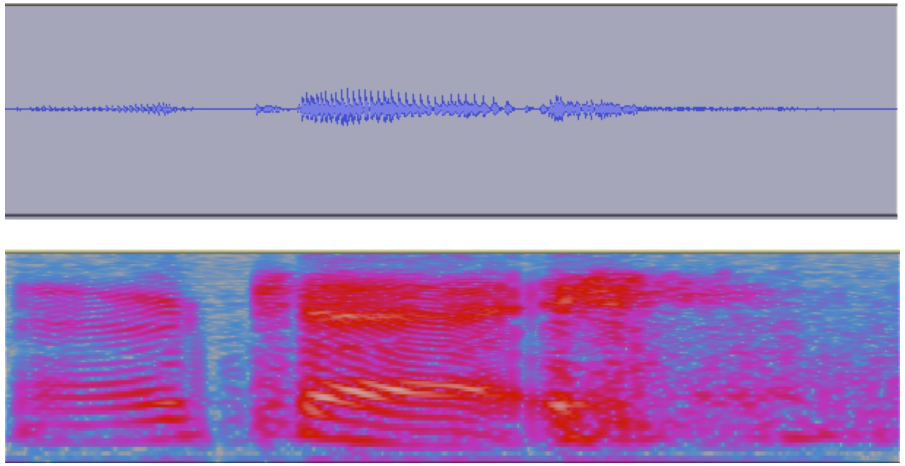
Waveform to speech feature



- Performed by so-called **feature extraction** module
  - Mel-frequency cepstral coefficient (MFCC), Perceptual Linear Prediction (PLP) used for **Gaussian mixture model (GMM)**
  - Log Mel filterbank used for **deep neural network (DNN)**
- Time scale
  - 0.0625 milliseconds (16kHz) to 10 milliseconds
- Type of values
  - Scalar (or discrete) to 12-dimensional vector
- **Mostly language-independent process**
  - Some languages use special features, e.g., pitch in Mandarin

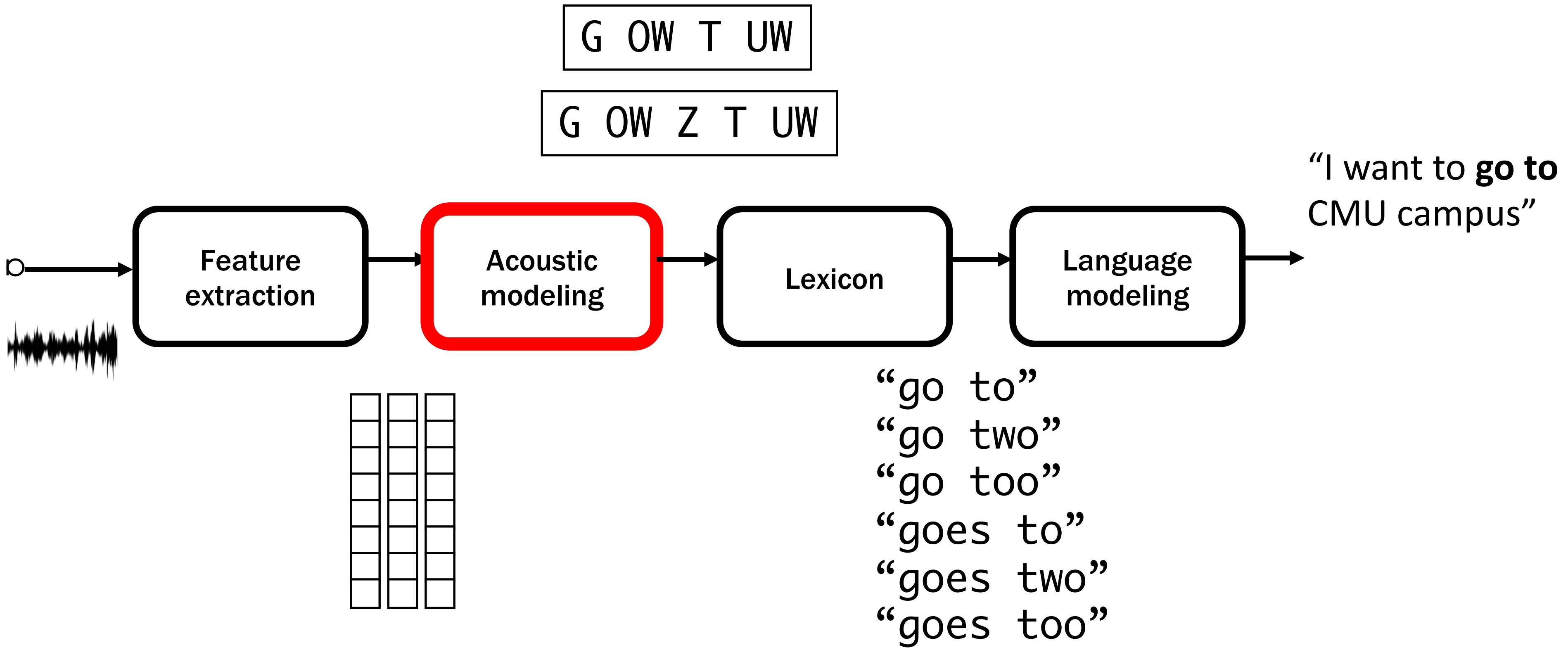
What kind of representation are desired?

- Need to preserve phonetic/linguistic information
- While suppressing irrelevant information (speakers and noises)
- Better to consider the compatibility with backend modules



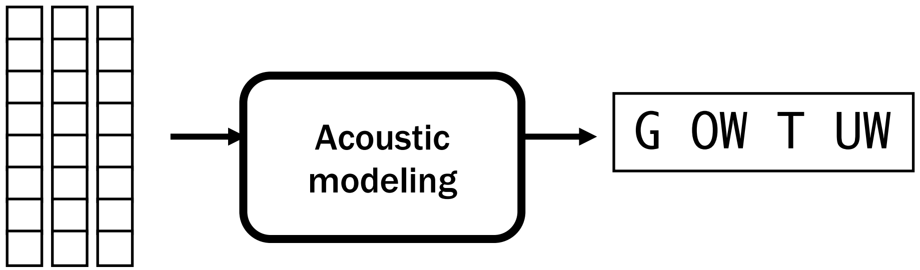
- Perceptual Linear Prediction (PLP) or multilayer perceptron tandem (MLP-Tandem)
- Learnable frontend (CNN)
- Self-supervised learning (HuBERT)

Speech recognition pipeline



$p(O|L)$ $p(L|W)$ $p(W)$

Speech feature to phoneme

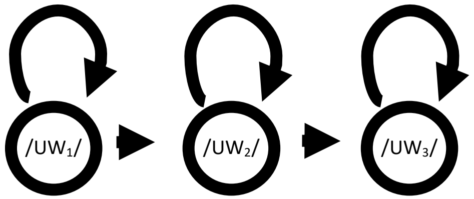


- Performed by so-called **acoustic modeling** module
  - Hidden Markov model (HMM) with **GMM** as an emission probability function
  - Hidden Markov model (HMM) with **DNN** as an emission probability function
- Time scale
  - 10 milliseconds to ~100 milliseconds (depending on a phoneme)
- Type of values
  - 12-dimensional continuous vector to 50 categorical value (~6bit)
- **Mostly language independent**
  - Map of the speech feature (language independent) to phoneme
- It can be a probability of possible phoneme sequences, e.g.,

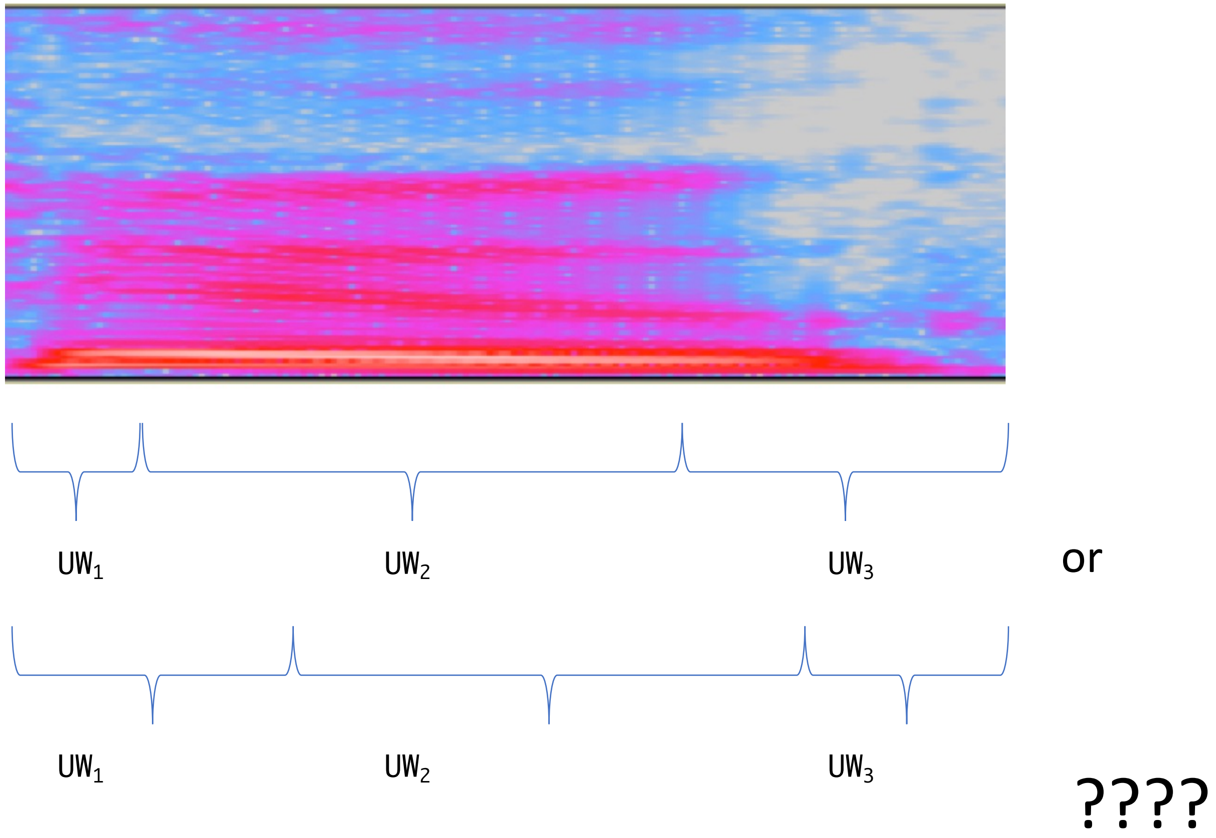
  G OW T UW or G OW Z T UW with some scores

Acoustic model $p(O|L)$

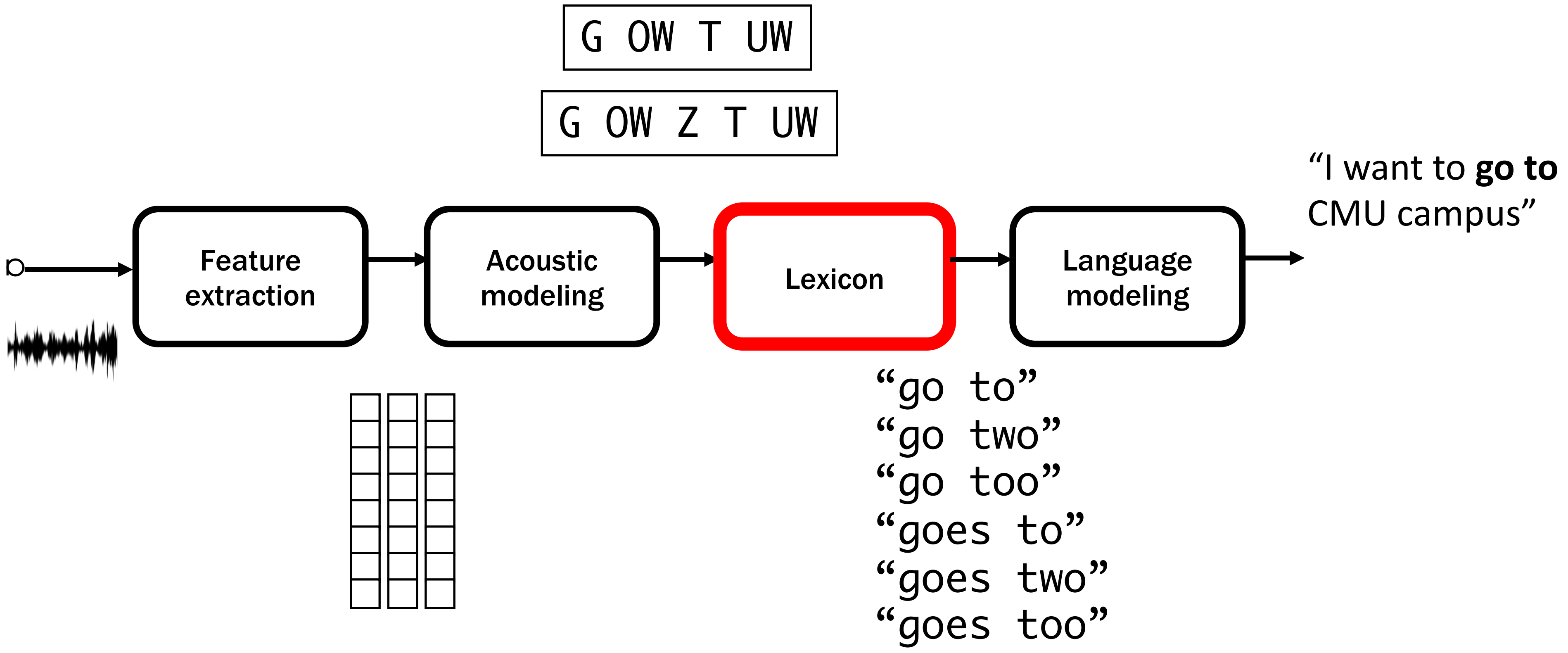
- O and L are **different lengths**
- **Align** speech features and phoneme sequences by using HMM



- Provide $p(O|L)$ based on this alignment and model
- The most important problem in speech recognition

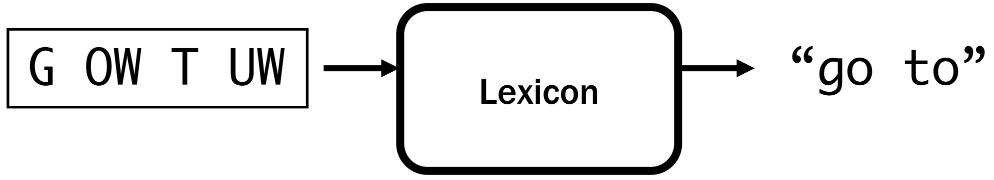


Speech recognition pipeline



$p(O|L)$ $p(L|W)$ $p(W)$

Phoneme to word



- Performed by **lexicon** module
  - American English: CMU dictionary
- Time scale
  - 100 milliseconds (depending on a phoneme) to 1 second (depending on a word and also language)
- Type of values
  - 50 categorical value (~6bit) to 100K categorical value (~2Byte)
- **Language dependent**
- It can be **multiple** word sequences (one to many)

Lexicon $p(L|W)$

- Basically use a pronunciation dictionary, and map a word to the corresponding phoneme sequence
  - with the probability = 1.0 when single pronunciation
  - with the probability = $1.0/J$ when multiple (J) pronunciations

$$p(L|W) = p(/\mathrm{T}/, /\mathrm{OW}/|\text{"two"}) = 1.0$$

What is phone and phoneme???
GO TO: “g oʊ t u” or “G OW T UW”

- Phone: g oʊ t u
  - Devised by International Phonetic Association
  - Physical categorization of speech sound
  - Not applicable to all languages, needs special characters, too many variations

- Phoneme: one of the units that distinguish one word from another in a **particular language**
  - /r/ and /l/ are degenerated in some languages (e.g., “rice” and “lice” sounds same for me!). Then, we don’t have to distinguish them.
  - ARPAbet: `G OW T UW`
    - Proposed by ARPA for the development of speech recognition of only “American English”
    - Represented by ASCII characters

Pronunciation dictionary

- CMU dictionary
  - http://www.speech.cs.cmu.edu/cgi-bin/cmudict

"I want to go to the CMU campus"

→AY W AA N T T UW G OW T UW DH AH S IY EH M Y UW K AE M P AH S

- Powerful, but limited
- Out of vocabulary issue, especially new word

→ Grapheme2Phoneme mapping based on machine learning

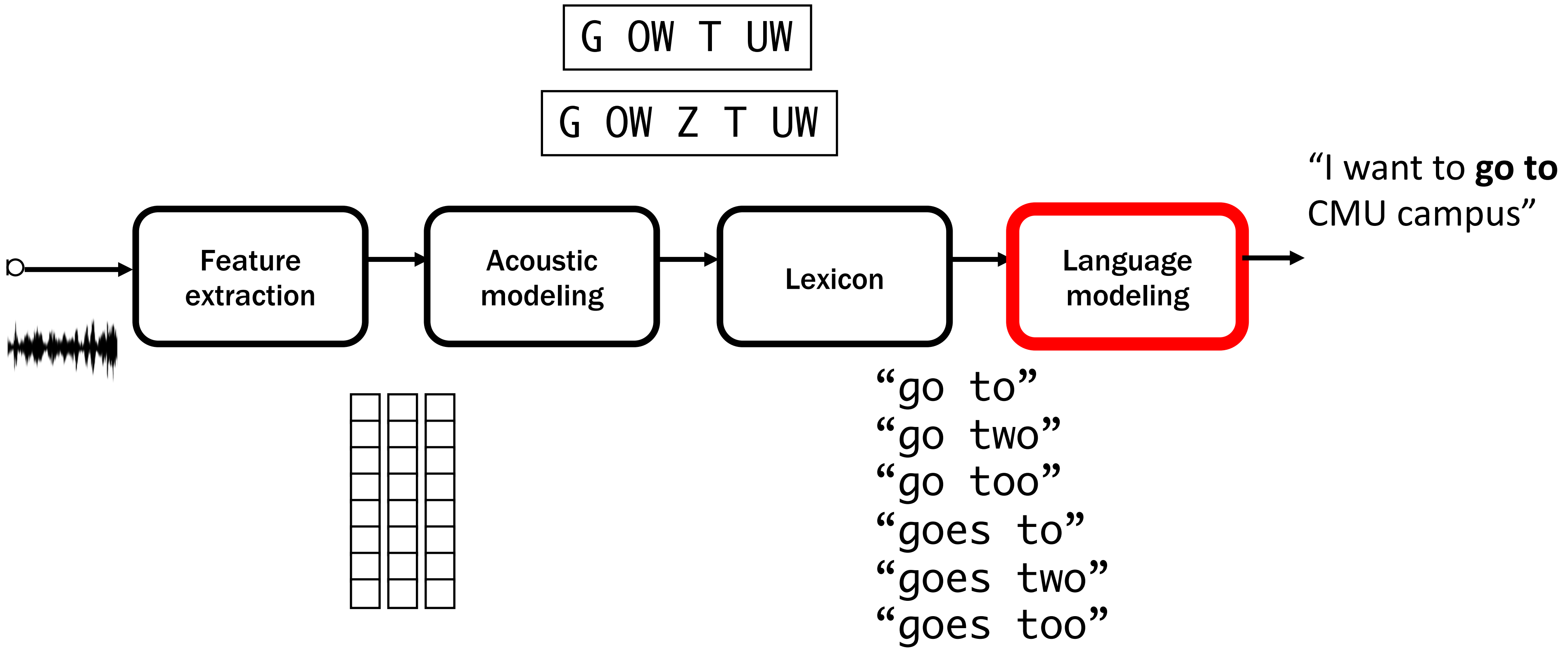
Let’s play the CMU dictionary!

- Access: http://www.speech.cs.cmu.edu/cgi-bin/cmudict
- Find some in-vocabulary words
- Find five out-of-vocabulary words

Multilingual phone dictionary

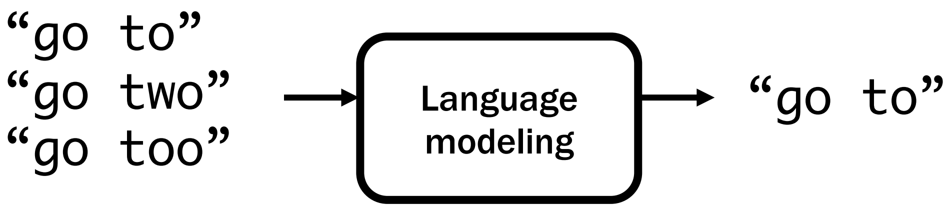
- https://en.wiktionary.org/wiki/Wiktionary:Main_Page

Speech recognition pipeline



$$p(O|L) \quad p(L|W) \quad p(W)$$

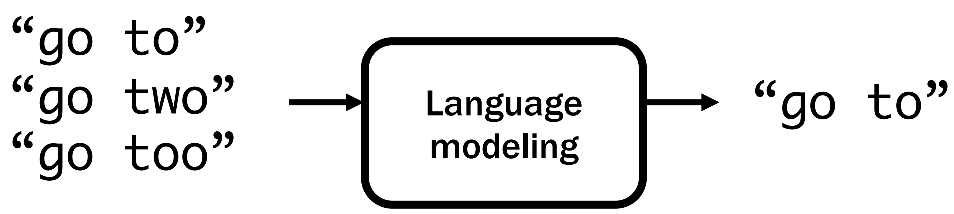
Word to text



- Performed by **language modeling** module $p(W)$
  - N-gram
  - Recurrent neural network language model (RNNLM)
- From training data, we can basically find how possibly "to", "two", and "too" will be appeared after "go"
  - Part of WSJ training data, 37,416 utterances
    - "go to": **51** times
    - "go two":
    - "go too":

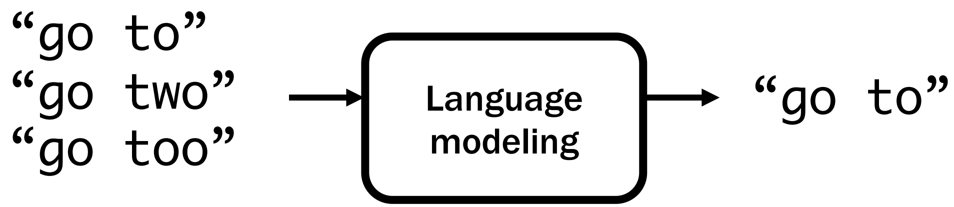
THE WALL STREET JOURNAL.

Word to text



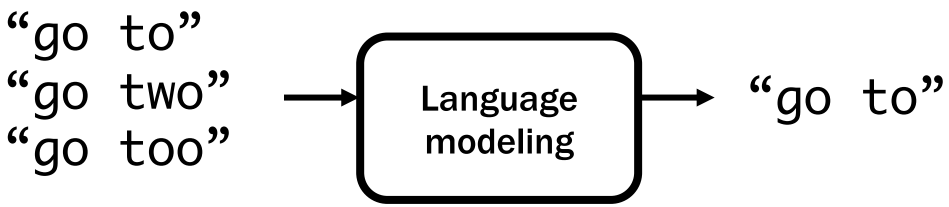
- Performed by **language modeling** module $p(W)$
  - N-gram
  - Recurrent neural network language model (RNNLM)
- From training data, we can basically find how possibly "to", "two", and "too" will be appeared after "go"
  - Part of WSJ training data, 37,416 utterances
    - "go to": **51** times
    - "go two": **0** times
    - "go too": **0** times

Word to text



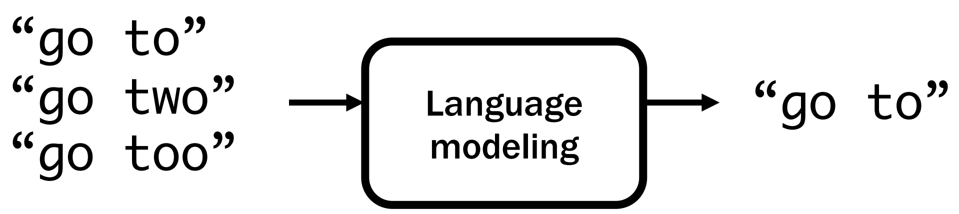
- Performed by **language modeling** module $p(W)$
  - N-gram
  - Recurrent neural network language model (RNNLM)
- From training data, we can basically find how possibly ”to”, “two”, and “too” will be appeared after “go”
  - WSJ all text data, 6,375,622 sentences
    - “go to”: **2710** times
    - “go two”:
    - “go too”:

Word to text



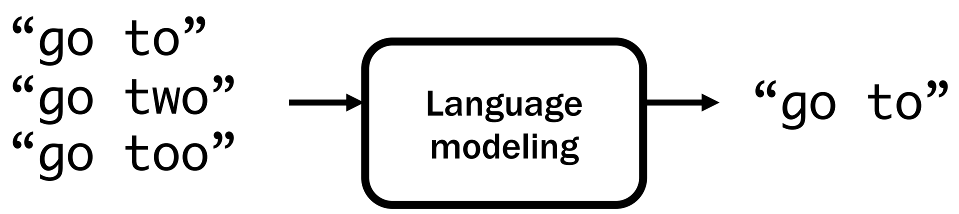
- Performed by **language modeling** module $p(W)$
  - N-gram
  - Recurrent neural network language model (RNNLM)
- From training data, we can basically find how possibly "to", "two", and "too" will be appeared after "go"
  - WSJ all text data, 6,375,622 sentences
    - "go to": **2710** times
    - "go two": **2** times, e.g., "those serving shore plants often go two hundred miles or more"
    - "go too":

Word to text



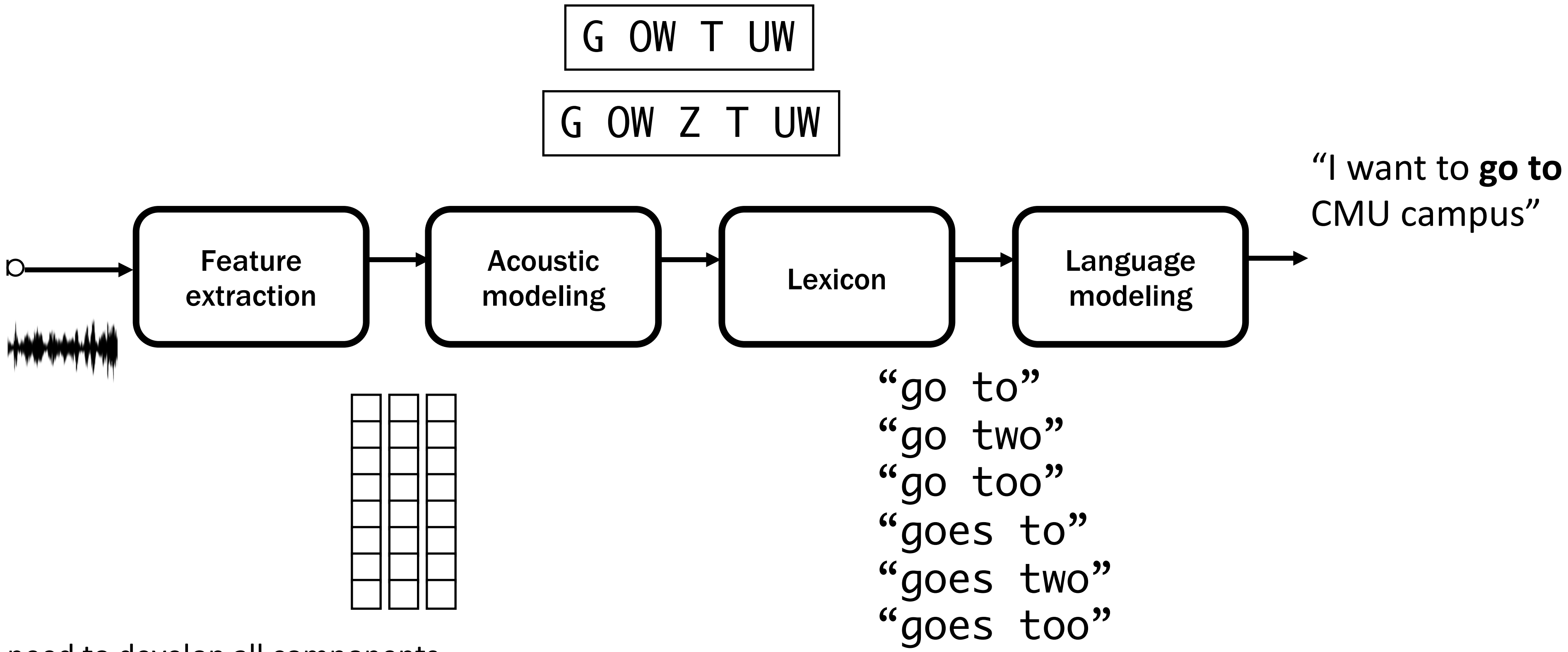
- Performed by **language modeling** module $p(W)$
  - N-gram
  - Recurrent neural network language model (RNNLM)
- From training data, we can basically find how possibly ”to”, “two”, and “too” will be appeared after “go”
  - WSJ all text data, 6,375,622 sentences
    - “go to”: **2710** times
    - “go two”: **2** times, e.g., “those serving shore plants often go two hundred miles or more”
    - “go too”: **41** times, e.g., “he could go too far”

Word to text



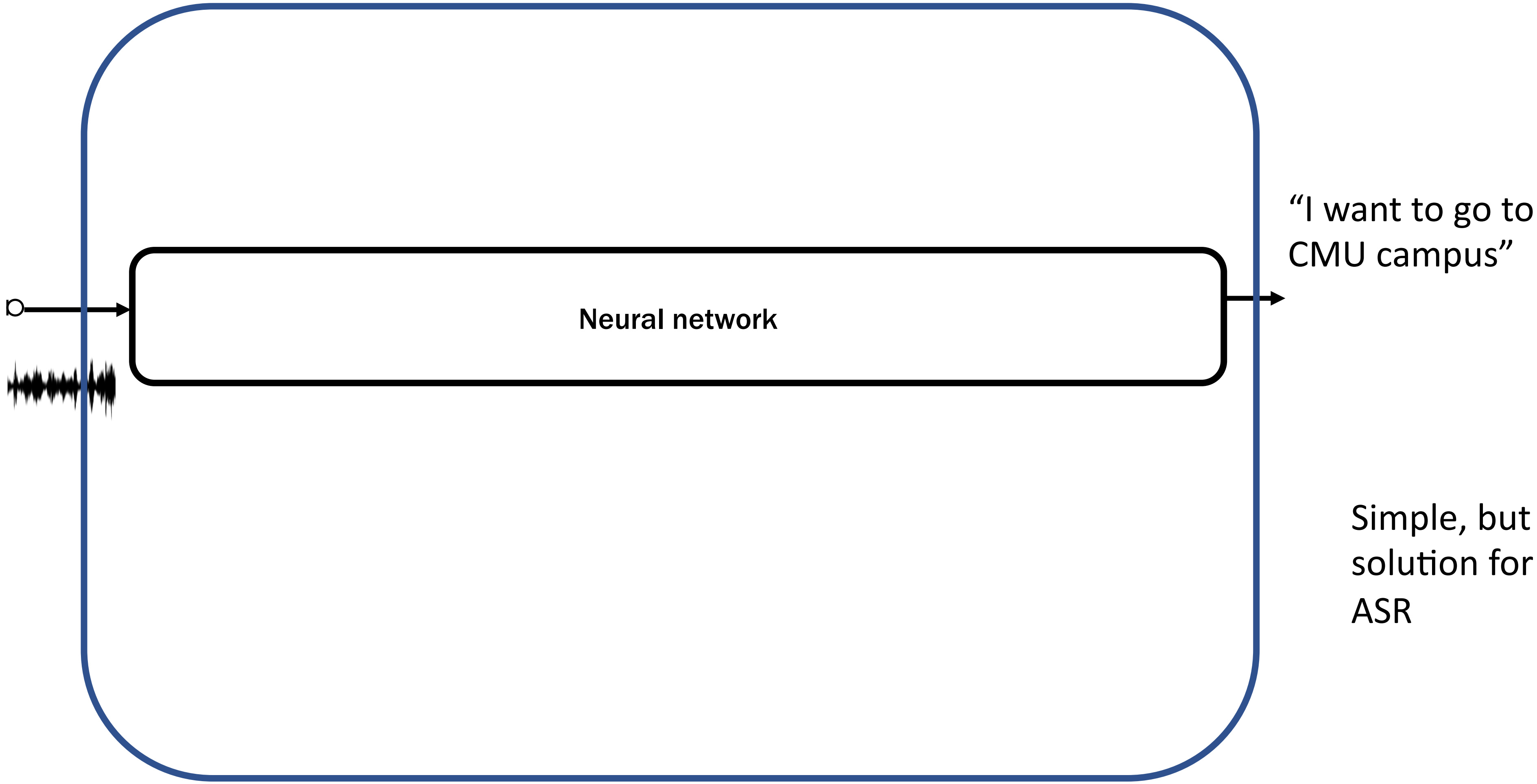
- Performed by **language modeling** module $p(W)$
  - N-gram
  - Recurrent neural network language model (RNNLM)
- From training data, we can basically find how possibly "to", "two", and "too" will be appeared after "go"
  - WSJ all text data, 6,375,622 sentences
    - "go to": **2710** times
    - "go two": **2** times, e.g., "those serving shore plants often go two hundred miles or more"
    - "go too": **41** times, e.g., "he could go too far"
- **Language dependent**

Building speech recognition is really difficult…



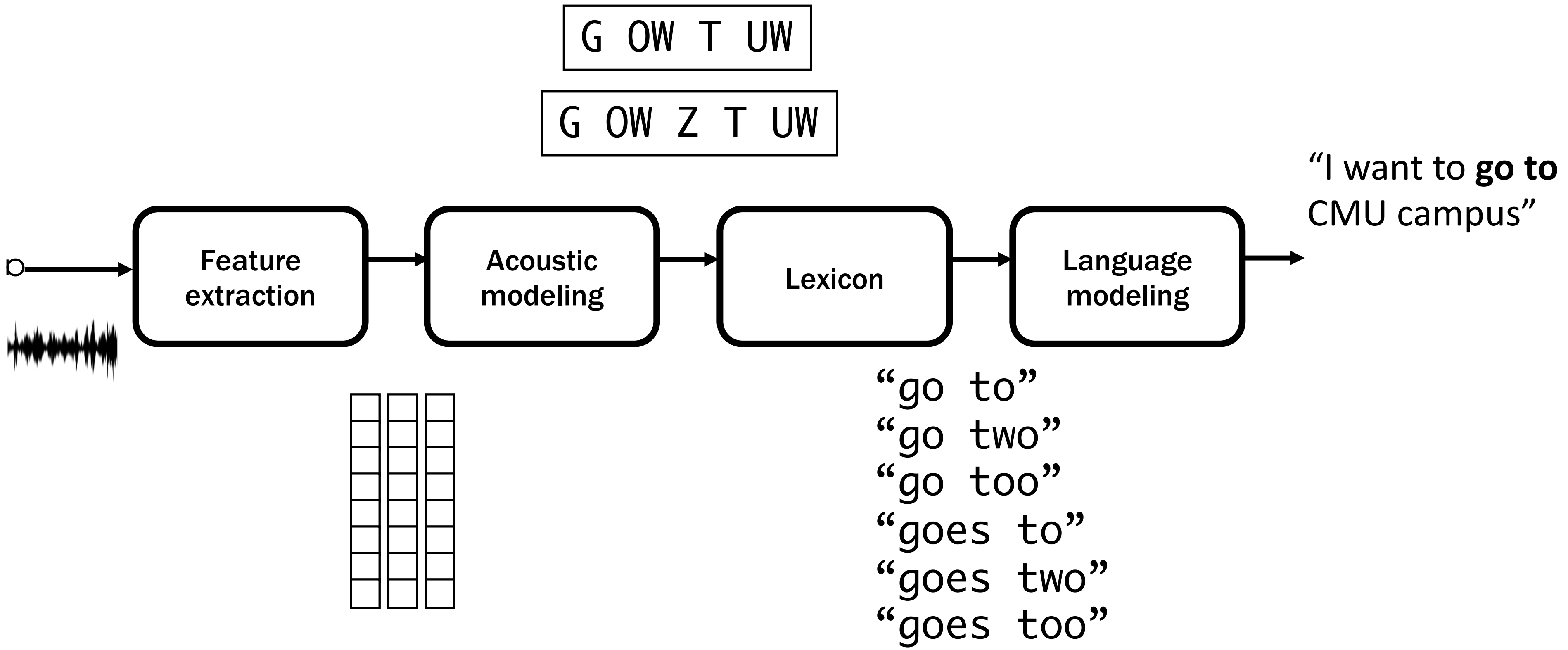
- We need to develop all components
- Each component requires a lot of background knowledge
- We need to tune hyper-parameters in each module

End-to-end speech recognition

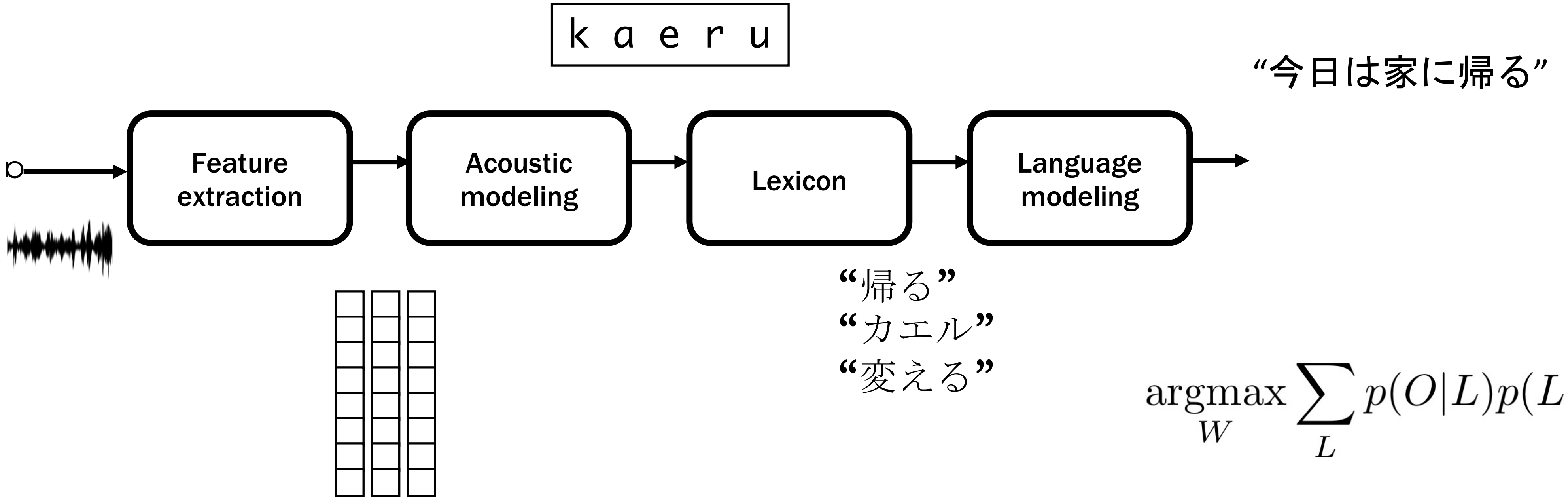


Simple, but not perfect solution for multilingual ASR

How to apply it to the other language?



How to apply it to the other language?



$$\underset{W}{\operatorname{argmax}} \sum_{L} p(O|L)p(L|W)p(W)$$

- We can just change the lexicon and language models (dictionary and text only data)
- Not easy for end-to-end ASR (we need parallel data)

Summary

- Speech recognition
  - Well defined problem (input: sound, output: text, evaluation metric)
  - The problem is well factorized with 1) feature extraction, 2) acoustic model, 3) lexicon, and 4) language model
  - 1 and 2 are mostly language intendent while 3 and 4 are language dependent

Discussion

1. Please try one of speech recognition engines in **English** (Google, Apple, Amazon, etc.)
   - If you find some recognition errors, please discuss why such errors occur
   - You can do some stress tests by making a difficult situation for the engine
2. Please try the above speech recognition engine in the **other language**
   - If you find some difference between English and the other language for the performance or behavior, please discuss it

In my case, I found that English and Japanese are equally good in terms of the ASR performance, but the spoken language understanding part in Japanese was very poor…