# Joint-Character-POC N-Gram Language Modeling For Chinese Speech Recognition

Bin Wang, Zhijian Ou

Speech Processing and Machine Intelligence (SPMI) Lab

Tsinghua University

Jian Li, Akinori Kawamura

Toshiba Corporation

# Summary

Introduction

Model definition

- Word n-gram language models
- Joint n-gram language models

Smoothing

- Revise the tradition smooth to suit joint situation.

Scoring

- WFST representation of joint n-gram LMs

Experiments

- For Chinese speech recognition

Related work and conclusion

# Introduction

The state-of-the-art language models (LMs) are word-based language models.

$$p(w_1, \ldots, w_l) = \prod_{i=1}^{l} p(w_i | w_{i-n+1}, \ldots, w_{i-1})$$

Drawbacks of word-based LMs

- The concept of word in Chinese is rather vague. There are no delimiters between adjacent Chinese words and even no standard definition of Chinese words.
- It is always possible to construct new words by combining multiple characters, which causes out-of-vocabulary (OOV) problem.

Contributions

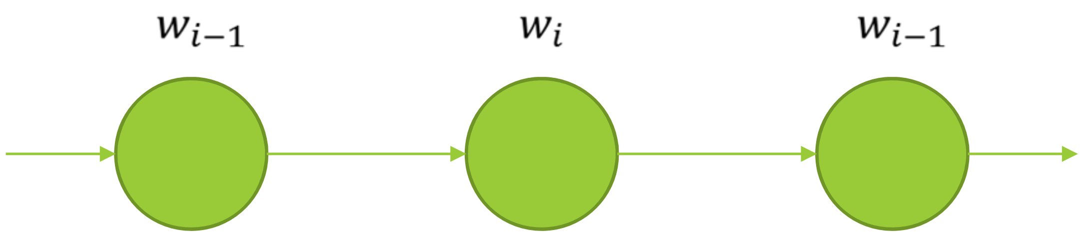
- Incorporate position-of-character (POC) tags into character-based n-gram models to model word-level and character-level constraints.
- Evaluate the performance on Chinese speech recognition, especially on OOV processing

# Word N-Gram Language Models

Word n-gram language model is one of the most popular language models, because of its quick estimation and well incorporation to the WFST-based 1-pass decoding, even the recent RNN language model has drawn much attention.

In word n-gram LMs, the basic unit is word. With the Markov assumption, current word only depends on previous n-1 words.

2-gram




The probability of a sentence is :

$$p(w_1, \dots, w_l) = \prod_{i=1}^{l} p(w_i | w_{i-n+1}, \dots, w_{i-1})$$

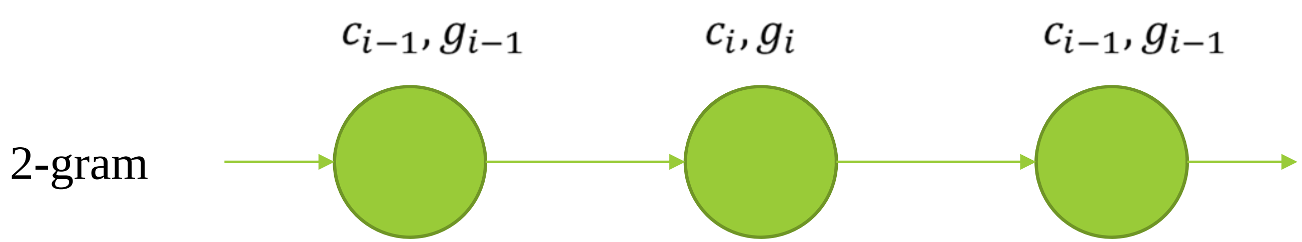
# Joint N-Gram LMs

Different with the word n-gram LMs, for joint n-gram LMs, the basic units are joint-states $(c_i, g_i)$, where $c_i$ denotes character and $g_i$ denotes POC tag.

Language modeling is essentially sequence modeling.

$$p(u_1, \dots, u_l) = \prod_{i=1}^{l} p(u_i | u_{i-n+1}, \dots, u_{i-1}) \qquad \begin{cases} u_i \leftarrow w_i & \text{word n-gram} \\ u_i \leftarrow c_i & \text{character n-gram} \\ u_i \leftarrow (c_i, g_i) & \text{joint n-gram} \end{cases}$$

The POC tag of a character can take 4 values – B, M, E and S, which represents the beginning, middle, end of a word and a single-character word respectively.



# Smoothing

Smoothing is used to overcome the sparseness problem of training corpus.

ML estimation

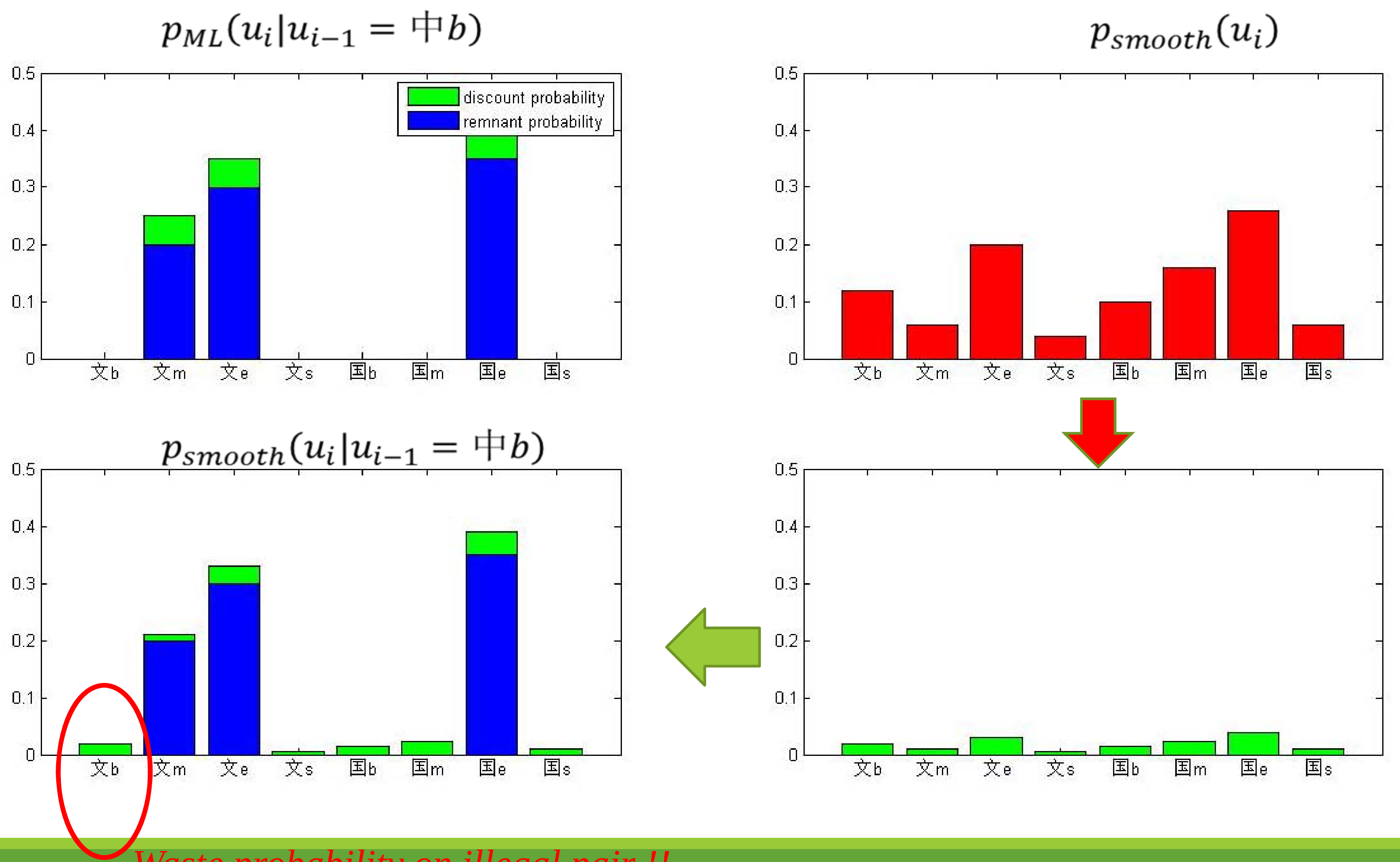
$$p_{ML}(u_i | u_{i-n+1}^{i-1}) = \frac{count(u_{i-n+1}^{i})}{count(u_{i-n+1}^{i-1})}$$

If $count(u_{i-n+1}^{i}) = 0$, then smoothing is required to avoid zero probabilities.

While for joint n-gram LMs, there are hard constraints between POC tags.

| POC tag $g_{i-1}$ | Following legal POC tag $g_i$ |
|---|---|
| B | M / E |
| M | M / E |
| E | B / S |
| S | B / S |

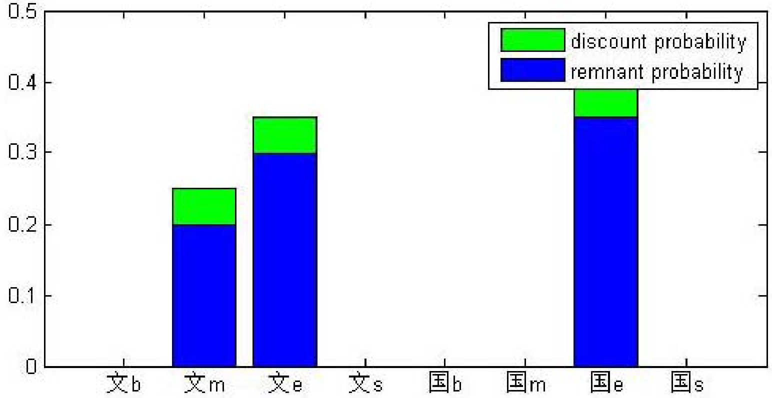
# Traditional Smoothing



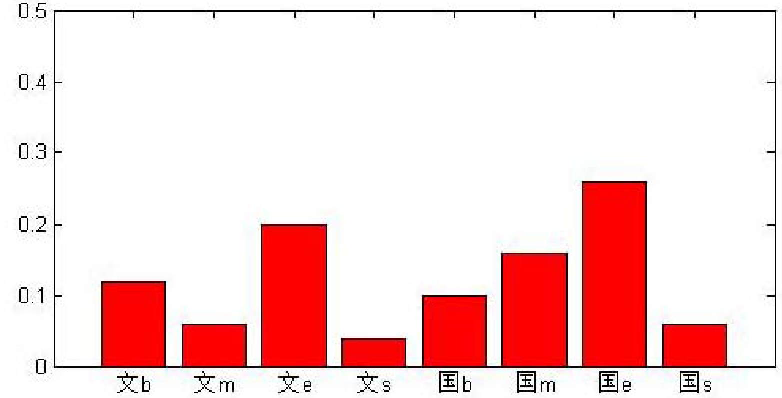

*Waste probability on illegal pair !!*

# Revised Smoothing

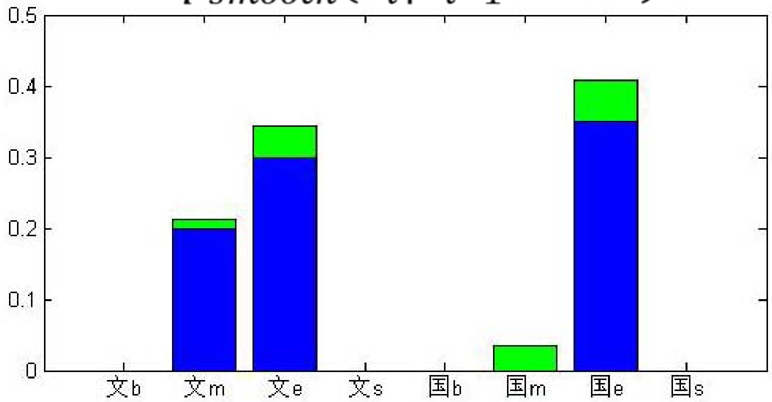
$p_{ML}(u_i|u_{i-1} = \text{中}b)$

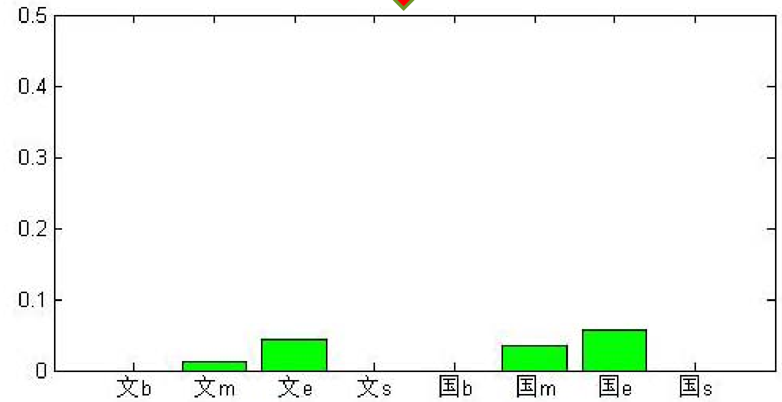



$p_{smooth}(u_i)$




$p_{smooth}(u_i|u_{i-1} = \text{中}b)$



# Scoring

Scoring is used to calculate the probability of a given character sequence. Because of the hidden variable, Viterbi approximation is often used to max-marginalize out the hidden variables, instead of the expensive sum-marginalization.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 欢 | 迎 | 参 | 观 | 清 | 华 | 大 | 学 |
| B | E | B | E | B | M | M | E |
| S | S | B | E | B | E | B | E |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

To get the probability of the given sentence, we need calculate the summation over all the possible POC tags.

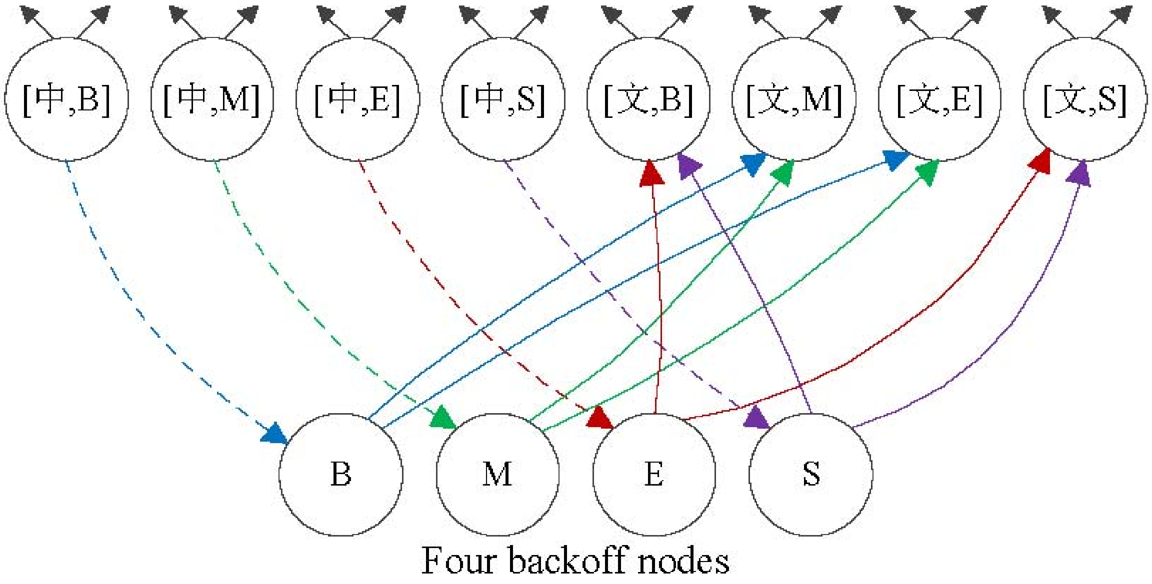
$$p(c_1^L) = \sum_{g_1^L} p([c,g]_1^L)$$

Viterbi approximation

$$p(c_1^L) \cong \max_{g_1^L} p([c,g]_1^L)$$

# Scoring

Representing LMs to WFSTs is an excellent way to perform Viterbi decoding. For word n-gram LMs, a standard algorithm for creating the WFST representation layer-by-layer has been introduced by C. Allauzen, M. Mohri and B. Roark. While after our revise, the WFST representation of joint n-gram should also be revised correspondingly.

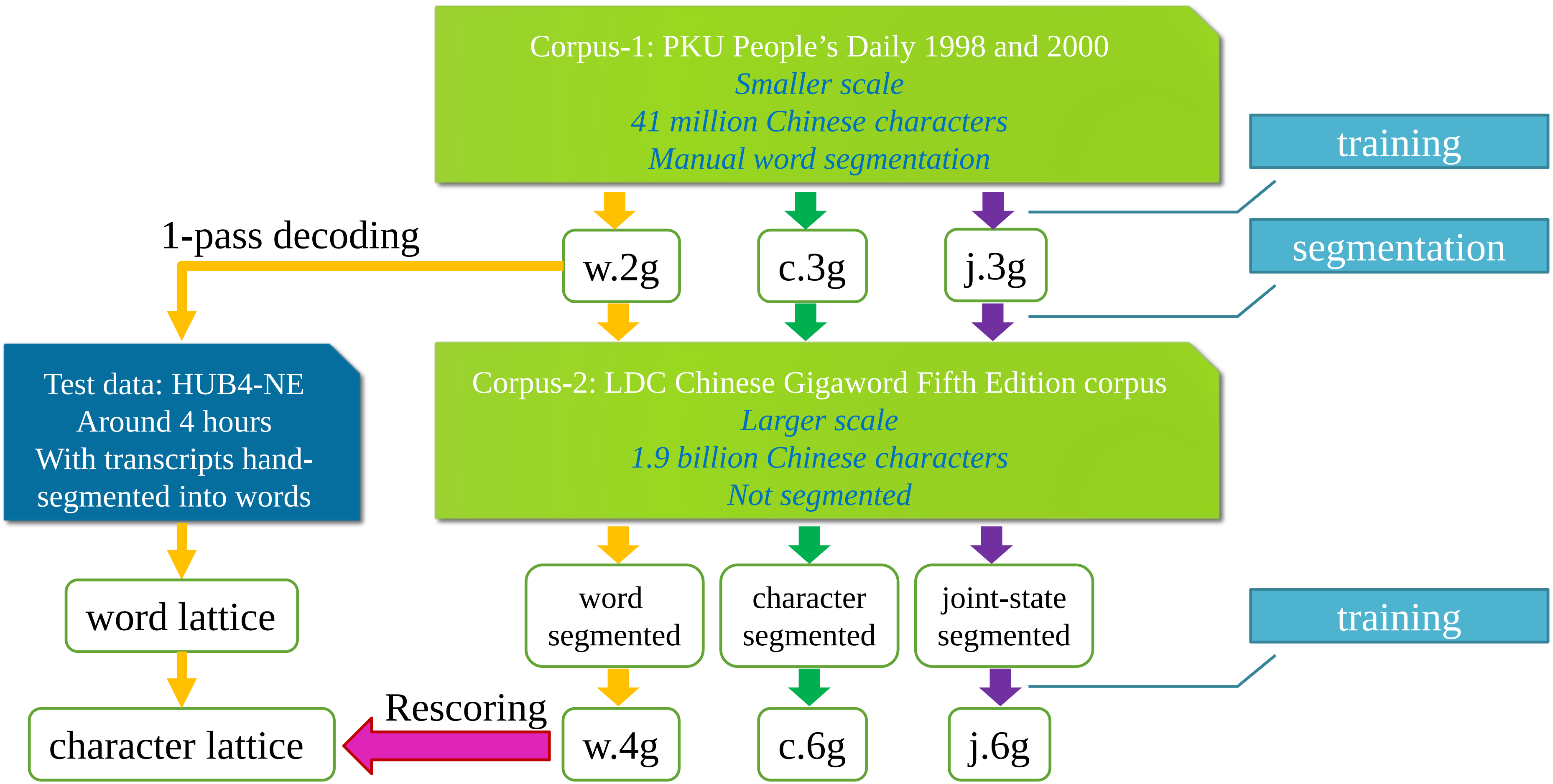



4 back-off nodes are used, corresponding to the 4 POC values, instead of the signal back-off node.

# Experiments

We test 3 kinds of LMs – word 4-gram, character 6-gram, joint 6-gram

*On average, one Chinese word contains 1.5 characters*



# Experiments

| | #state | #n-gram | cut-off setting | Perplexity | Error rates (%) | | |
|---|---|---|---|---|---|---|---|
| | | | | | CER | OOV-utt-CER | IV-utt-CER |
| Oracle | – | – | – | – | 4.77 | 6.66 | 4.06 |
| w.4g | 58,916 | 130,118,547 | 0-0-1-3 | 28.43 | 20.98 | 23.26 | 20.13 |
| c.6g | 5,032 | 274,544,846 | 0-0-0-1-1-3 | 29.01 | 20.86 | 23.18 | 20.00 |
| j.6g | 15,340 | 299,239,752 | 0-0-0-1-1-3 | 28.71 | 20.84 | 22.83 | 20.10 |
| w.4g∘c.6g | – | – | – | – | 20.58 | 22.79 | 19.76 |
| w.4g∘j.6g | – | – | – | – | 20.65 | 22.74 | 19.87 |

Table 1: Perplexities and error rates for different LMs. #states represents the number of words, characters and joint-states respectively for w.4g, c.6g and j.6g. #n-gram represents the total number of n-grams of all orders. As an example of the terminology we use to describe cut-off settings, 0-0-1-3 means that all unigrams with 0 or fewer counts are ignored, all bigrams with 0 or fewer counts are ignored, all trigrams with 1 or fewer counts are ignored, and all fourgrams with 3 or fewer counts are ignored.

**OOV-utt-CER**: *the CER on OOV-utterance subset, in which each utterance contains at least one OOV word.*
**IV-utt-CER**: *the CER on IV-utterance subset, in which all the words are in-vocabulary (IV) words.*

j.6g shows the advantage in recognizing OOV utterance, and gains 1.8% relative reduction of OOV-utt-CER compared to w.4g

# Related Work

**Neural network LMs – Feedforward NNLMs and recurrent NNLMs**

- To embed words into a continuous space in which probabilities are computed via smooth functions implemented by neural networks.
- To address the problem of data sparseness and achieve better generalization for unseen n-grams.

**Feature-based LMs**

- Such as class-based LMs and factored LMs
- Successfully used in morphologically rich European languages to overcome OOV problem.

**Our motivation is mainly linguistically-inspired**

- To explore both word-level and character-level constraints.
- To address the OOV problem for Chinese LMs.

*It can been seen from the experiments that the performance of joint LMs may be limited by sparse estimation of the parameters. Therefore it is interesting to find better smoothing method.*

Thanks for your attention