



Phoneme-based Transliteration of Foreign Names for OOV Problem

Wei Gao, Kam-Fai Wong and Wai Lam

Department of Systems Engineering & Engineering Management

The Chinese University of Hong Kong

{wgao, kfwong, wlam}@se.cuhk.edu.hk

Talk Outline

- Brief overview of OOV problem and Machine Transliteration in cross language applications
 - Problems of existing models in English-Chinese transliteration task
 - Fully handcrafted rule based model (Wan and Verspoor, 1998, *ACL*)
 - Handcrafted phonological rules plus automatic generation of phonetic mapping rules (Meng et al., 2001, *ASRU*)
 - Fully data-driven model based on Source-Channel/IBM MT (Virga and Khudanpur, 2003, *ACL workshop on Multi-lingual NER*)
 - A statistical transliteration a method based on direct estimation of phonetic symbol-mapping plus target language model
 - Experiments and Discussions
 - Conclusions
-

Machine Transliteration Overview

- Automatic transliteration of foreign names (of people, places, organizations, etc) — an important issue in cross language applications. E.g. CLIR, MT, SLP...
- Proper name dictionaries can never be comprehensive rendering name translation ineffective — OOV (Out-Of-Vocabulary) problem.
- How are foreign names translated by people?
 - Rule of thumb
 - Defacto standard, no absolute standard
 - Dialectical properties: Bush—布什/布殊/布希; Hussein—侯赛因/海珊/哈珊
bu shi bu shu bu xi hou sai yin hai shan ha shan
- Machine Transliteration:
 - forward/backward;
 - phoneme-based/grapheme-based;
 - rule-based/statistical

Related Work—Fully handcraft rule base (Wan and Verspoor, 1998, *ACL/COLING*)

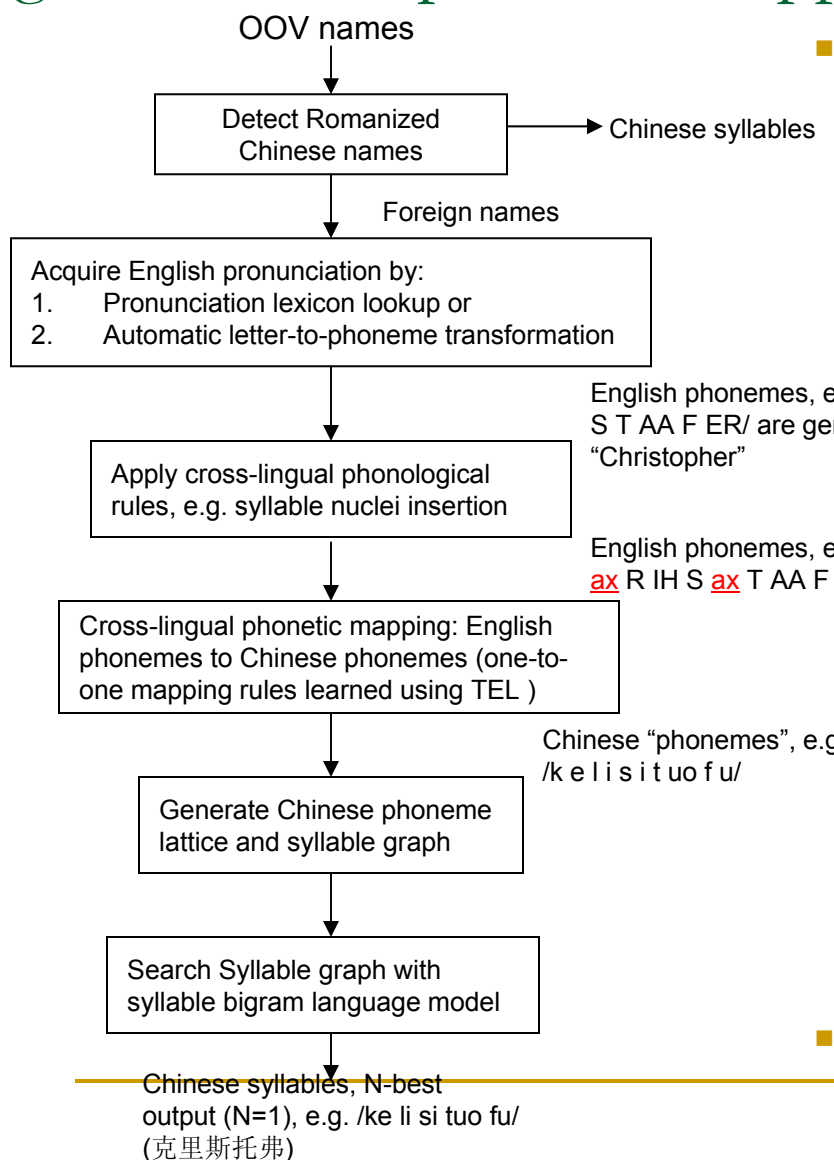
	f-	n-	p-	r-	v
a	fa	na	ba	la	wa
ae	fei	nei	bei	lei	wei
ai	fei	nei	bei	lei	wei
ai	fai	nai	bai	lai	wai
ai	fa yi	na yi	ba yi	la yi	wa yi
ao	-	nao	bao	lao	-
ar#	-	nuo	-	luo	wuo
au	-	nuo	-	luo	wuo
ay	fei	nei	bei	lei	wei
o	fo	-	bo	-	wo
o#	-	nuo#	-	luo#	wuo#
oa	-	-	bo ya	-	wo ya
oe	-	nuo	-	luo	wuo
oi	-	-	-	-	#
on	-	-	-	lun	-
or#	-	nuo#	-	luo#	wuo#
ou	-	nuo	-	luo	wuo

Table 1. Portion of handcrafted English phoneme-Chinese Pinyin Mapping Table

Transliteration of “Faeroe Islands” is as follows:

1. No match for “Faeroe” in the dictionary, so must be transliterated;
2. Divide “Faeroe” into two syllables by recognizing the syllabic break falls before the ‘r’ in the middle of consonant group;
3. Map /fae/ and /roe/ onto their Chinese equivalents. Since no vowel form /ae/ exists in Chinese, this is mapped to /ei/. The /r/ of the second syllable is mapped to // and /oe/ is correspondingly mapped to /uo/;
4. Since each syllable is of the form <cv>, no subsyllabic processing is required;
5. The transliterated phrase “fei luo” is mapped to Han characters: “非罗”
6. “Island” is lexically translated into “群岛”.

Related Work—Handcraft phonological rules plus automatic generation of phonetic mapping rules (Meng et al., 2001, *ASRU*)



- Three cross-lingual phonological rules can't bridge all the gaps:
 - Insert a reduced nuclei /ax/ between clustered consonants.
 - E.g. Clinton -> /K L IH N T IH N/ -> /K ax L IH N T IH N/ -> /k e l i n d u n/ -> 克林顿;
 - **Failure on:** Forest -> /F AO R AH S T/ -> /f u l e i/ -> 福雷; Birmingham -> /B ER M IH NG HH AE M/ -> /b o m i n g h a n/ -> 伯明翰
 - Duplicate nasals /M/, /N/, etc. whenever they are surrounded by vowels.
 - E.g. Diana -> /D AY AE N AH/ -> /D AY AE N N AH/ -> /d ai a n n a/ -> 戴安娜;
 - **Failure on:** Canada -> /K AE N AH D AH/ -> /j ia n a d a/ -> 加拿大; Havana -> /HH AH V AE N AH/ -> /h a v a n a/ -> 哈瓦纳
 - For all consonant endings, except /L/, append a syllable nuclei /ax/ to it.
 - E.g. Bennett -> /B EH N IH T/ -> /B EH N IH T ax/ -> /b e i n e i t e/ -> 贝内特;
 - **Failure on:** Auckland -> /AA K L AH N D/ -> /a o k e l a n/ -> 奥克兰; Alford -> /AE L F ER D/ -> /a e r f u/ -> 阿尔福
- Unable to transform the inserted or appended /ax/ to correct target phoneme during cross-lingual phonetic mapping.

Related Work—IBM’s MT model based on source-channel (Virga and Khudanpur, 2003, *ACL workshop for NER*)

English Name	FRANCES TAYLOR						
English Phonemes	F	R	AE	NS	IHS	TEY	LER
Initials and <u>Finals</u>	f	u	l	ang	x	i	s
Chinese Pinyin	fu	lang	xi	si	tai	le	
Chinese Transliteration	弗	朗	西	丝	泰	勒	

Figure 1. English-to-Chinese transliteration example
(Virga and Khudanpur, 2003)

IBM’s MT model based on Bayes rule:

$$\hat{C} = \arg \max_c p(C | E) = \arg \max_c p(E | C)p(C)$$

$$p(E | C) = \sum_A p(E, A | C) \approx p(E, \hat{A} | C)$$

Estimated using Bootstrapping: EM iterations
of Model-1 + Model-2 + HMM + Model-4

$$p(C) = \prod_{i=1}^{|C|} p(c_i | c_{i-2}c_{i-1})$$

Pinyin trigram + Good-Turing smoothing

- Note that the component of $p(E|C)$ in source-channel is estimated with the opposite direction of actual E-to-C transliteration.
- Unrealistic constraints on mapping fertility: allows only a target language phoneme to be associated with a contiguous group of source language phonemes, but not vice versa, i.e. one English phoneme can never be converted to a group of pinyin symbols.
- Zero fertility symbols: /u/, /i/ are spuriously generated from nowhere. They are “deleted” by source-channel during training, but need to be “reproduced” with certain probability $1/\phi!$ by channel decoder: ϕ is the number of zero fertility symbols found {i, e, u, o, r, v, ou, c, u, ie, etc.}

A New Transliteration Model—direct estimation on symbol-mappings plus target language model

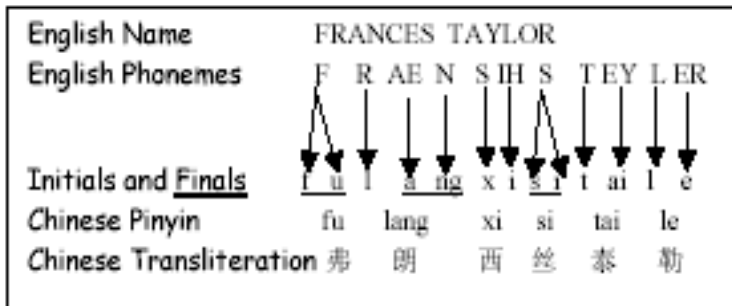


Figure 2. Our phoneme alignment scheme in direct transliteration modeling

- Mathematical Soundness of direct model under MaxEnt framework (Och and Ney, 2002, ACL)

$$p(C | E) = p_{\lambda_1^M}(C | E) = \frac{\exp \left[\sum_{m=1}^M \lambda_m h_m(E, C) \right]}{\sum_{E'} \exp \left[\sum_{m=1}^M \lambda_m h_m(E', C) \right]}$$

$$\hat{C} = \arg \max_C p(C | E) = \arg \max_C \left\{ \exp \left[\sum_{m=1}^M \lambda_m h_m(E, C) \right] \right\}$$

We set:

$$h_1(E, C) = \log p_{\theta}(C | E) \quad h_1(E, C) = \log p_{\theta}(E | C)$$

$$h_2(E, C) = \log p_{\gamma}(C) \quad h_2(E, C) = \log p_{\gamma}(C)$$

$$\lambda_1 = \lambda_2 = 1$$

$$\lambda_1 = \lambda_2 = 1$$

It has to be empirically verify which set of features yields better results.

$$\hat{C} = ? \arg \max_C p(C | E) p(C)$$

$$p(C | E) = \sum_A p(C, A | E) \approx p(C, \hat{A} | E)$$

$$p(C, \hat{A} | E) = \sum_{i=1}^{|E|} p(\text{cmu}_i | e_i)$$

cmu (**mapping units** in pinyin) could be individual pinyin symbol or initial-final cluster converted from single English phoneme

Phonetic Alphabets

English: ARPABET (IPA in ASCII):

24 consonants	P	T	K	B	D	G	M	N	NG	F	V	TH
	DH	S	Z	SH	ZH	CH	JH	L	W	R	Y	H
16 vowels	IY	IH	EY	EH	AE	ER	UH	AX	AY	AW	AA	OW
	OY	AO	UW	UH								

Chinese: Pinyin (Initials & Finals)

23 initials	b	p	m	f	d	t	n	l	g	k	h	j
	q	x	zh	ch	sh	r	z	c	s	y	w	
35 finals	a	o	e	ai	ei	ao	ou	er	an	en	ang	eng
	i	ia	ie	iao	iu	ian	in	iang	ing	iong	u	ua
	uo	uai	ui	uan	un	uang	ong	v	ve	van	un	

- Considering an individual English phoneme, 4 general mapping possibilities exist:
 - It maps to an initial or a final (most common case);
 - It maps to an initial-final cluster, e.g. /F/ - /fu/ and /S/ - /si/ in previous example;
 - It maps to a dumb sound ϵ , e.g. /S T AE N F ER D/ (Stanford) to /si tan fu/ (斯坦福), where /D/ is omitted in translation.
 - Insert additional pinyin syllables, the equivalent to **REAL** zero fertility, e.g. /AE L B AH K ER K IY/ (Albuquerque) to /a er bo ke er ji/ (阿尔伯克尔基), where the second /er/ is excessively produced.

Preprocessing: Alignment of phoneme chunks

- Identifying a set of indicative sound units (indicators):
 - For an English phoneme sequence, they are:
 - All the consonants;
 - Vowel at the first position;
 - The second vowel of two contiguous vowels.
 - For a pinyin symbol sequence, they are:
 - All the initials;
 - Final at the first position;
 - The second final of two contiguous finals.
 - *Similar set of indicators exist in other romanization systems in Chinese and they are independent of alignment model
- Related variables:
 - $\tau(S) = \#$ of indicators in a sequence S .
 - $t = \max\{\tau(E), \tau(C)\}$
 - $d = |\tau(E) - \tau(C)|$

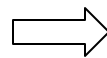
Preprocessing: Alignment of phoneme chunks (cont')

- Chunk E and C by tagging indicators identified;
- Compensate the one with fewer indicators by inserting d dumb sound ε at its $\min\{\tau(E), \tau(C)\}$ possible positions ahead of existing indicators;
- ε is considered as an indicator participating alignment so that two sequences both have as many as t indicators;
- The t chunks separated by indicators in E should align to the corresponding t chunks in C in the same order;
- There are $\|A\| = C_t^d = \binom{P_t^d}{d!} = \frac{t!}{(t-d)!d!}$ number of possible alignments at chunk-level with respect to different positions of ε .
- The method can guarantee each chunk contains two sound units at most. In a pair of aligned chunks, only three mapping layouts between elements are possible:
 - e-to-c1c2: c1c2 is considered as a cluster (initial-final);
 - e1e2-to-c1c2: e1-to-c1 and e2-to-c2;
 - e1e2-to-c: Add an additional ε at C side; e1-to-c and e2-to- ε or e1-to- ε and e2-to-c. $\|A\| = \|A\| + 1$;

EM Training for Symbol Mapping and Viterbi

Alignment

/AE /L /B AH /K ER /K IY
/a /er /b o /k e /er /j i



/ε /AE /L /B AH /K ER /K IY
/a /er /b o /k e /er /j i

/AE /ε /L /B AH /K ER /K IY
/a /er /b o /k e /er /j i

/AE /L /ε /B AH /K ER /K IY
/a /er /b o /k e /er /j i

/AE /L /B AH /ε /K ER /K IY
/a /er /b o /k e /er /j i

/AE /L /B AH /K ER /ε /K IY
/a /er /b o /k e /er /j i

5 possible alignments at chunk level, but 9 at phoneme level

EM Training:

- 1). Initialization: For each English-Chinese pair, assign equal weights to all initial alignments generated as $|A|^{-1}$.
- 2). Expectation Step: For each of the 40 English phonemes, count up instances of its different mappings from the observations on all alignments produced. Each alignment contributes counts in proportion to its own weight. Normalize the scores the mappings.
- 3). Maximization Step: Recompute the alignment scores. Each alignment is scored with the product of the scores of the symbol mappings it contains. Normalize the alignment scores.
- 4). Repeat step 2-3 until the symbol-mapping probabilities converge, meaning that the variation of each probability between two iterations becomes less than a specified threshold.

A1	A2	A3	A4
0.25	0.25	0.25	0.25

e	c	p(c e)	c	p(c e)	c	p(c e)	c	p(c e)
AA	ao	0.625	a	0.272	o	0.100	e	0.003
AE	a	0.519	ai	0.362	ao	0.143		
...	

A1	A2	A3	A4
0.54	0.13	0.11	0.22

EM Training Results

WFST for Phonetic Transition

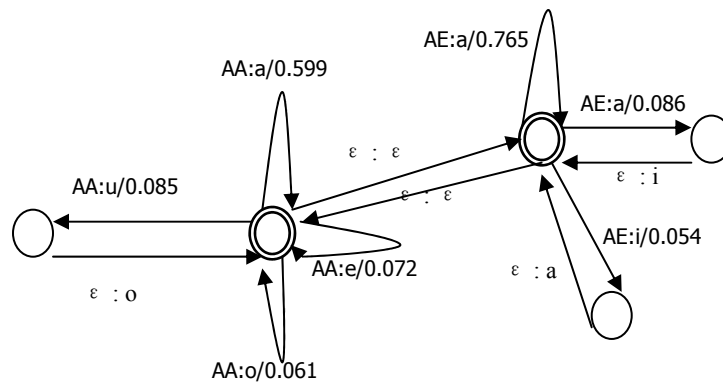


Figure 4. Part of WFST based on $P(C|E)$

FST for Legal Syllables

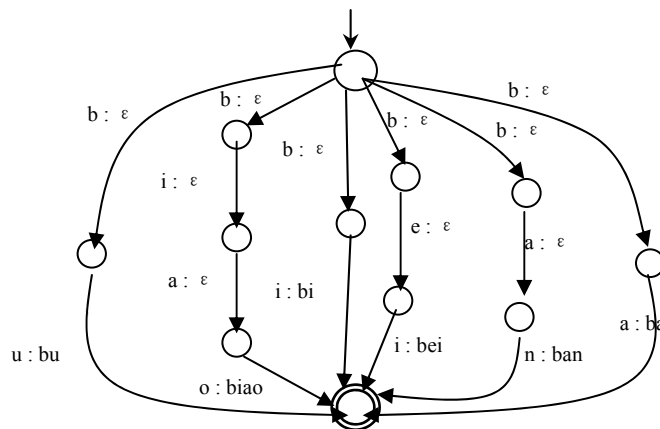


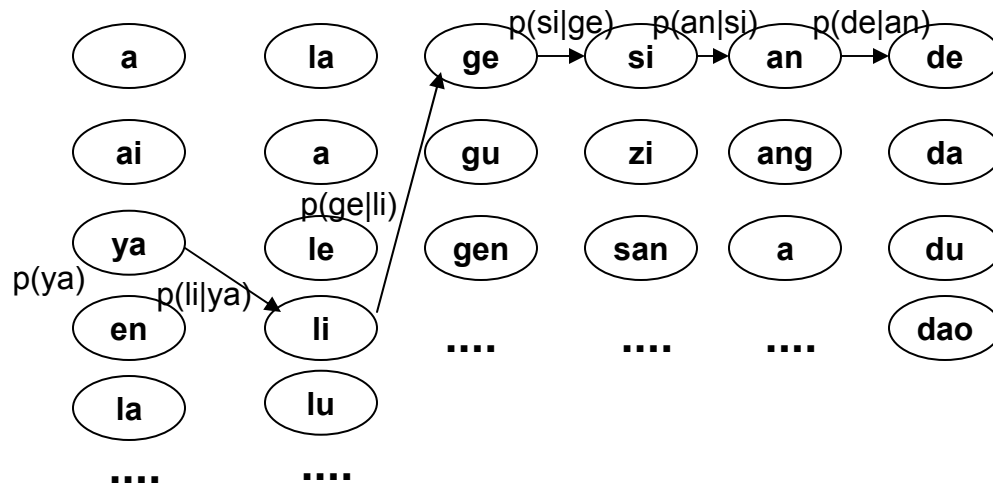
Figure 5. Part of FST for pinyin correction

e	c	$P(c e)$	c	$P(c e)$	c	$P(c e)$	c	$P(c e)$
AA	a	0.599	uo	0.085	e	0.072	o	0.061
AE	a	0.765	ai	0.086	ia	0.054	ya	0.049
AH	a	0.344	i	0.137	u	0.119	e	0.099
AO	uo	0.246	o	0.203	ao	0.174	e	0.159
AW	u	0.643	a	0.214	ei	0.071	uo	0.071
AY	ai	0.612	i	0.163	a	0.069	ei	0.041
B	b	0.737	bu	0.178	bo	0.016	p	0.016
CH	q	0.355	ch	0.290	j	0.129	sh	0.065
D	d	0.623	de	0.202	e	0.116	t	0.022
DH	si	1.000						
EH	ai	0.312	ei	0.174	e	0.123	a	0.065
ER	e	0.263	o	0.225	a	0.116	u	0.093
EY	ai	0.321	a	0.256	ei	0.231	i	0.077
F	f	0.625	fu	0.264	fo	0.069	fa	0.028
G	g	0.407	ge	0.241	j	0.1667	e	0.111
HH	h	0.810	x	0.114	xiu	0.038	e	0.025
IH	i	0.581	ei	0.156	yi	0.138	e	0.031
IY	i	0.696	ei	0.094	e	0.060	yi	0.045
JH	j	0.243	v	0.186	zh	0.171	q	0.114
K	k	0.468	ke	0.193	j	0.154	g	0.063
L	l	0.553	er	0.354	e	0.054	li	0.016
M	m	0.590	mu	0.137	n	0.137	ng	0.085
N	n	0.655	ng	0.160	e	0.125	en	0.027
NG	n	0.550	ng	0.450				
OW	uo	0.287	e	0.278	o	0.148	e	0.055
OY	uo	0.429	o	0.286	ei	0.143	v	0.142
P	b	0.324	p	0.297	pu	0.270	e	0.054
R	l	0.565	e	0.312	er	0.082	r	0.016
S	si	0.523	s	0.166	x	0.080	e	0.057
SH	sh	0.294	shi	0.176	x	0.176	sai	0.059
T	d	0.335	t	0.254	te	0.243	e	0.115
TH	s	0.393	si	0.250	e	0.077	d	0.071
UH	u	0.882	ou	0.059	e	0.059		
UW	u	0.609	e	0.132	ou	0.056	o	0.037
V	w	0.759	f	0.148	fu	0.074	wei	0.019
W	w	0.586	a	0.138	h	0.137	e	0.034
Y	y	0.500	e	0.213	k	0.125	n	0.083
Z	si	0.483	s	0.115	e	0.102	zi	0.092
ZH	x	0.750	r	0.250				
ε	n	0.257	yi	0.121	er	0.106	a	0.099

Table 2. English Phonemes with probabilistic mappings to Chinese pinyin sound units

Postprocessing: Using Bigram Language Model to Search N-best Transliterations

- Search syllable graph AE L AH G Z AE N D ER (Alexander) /ya li ge si an de/ (亚里格斯安德)



$$p(C) \cong \prod_i p(c_i | c_{i-1}) \cong \prod_i \frac{\text{count}(c_{i-1}c_i)}{\text{count}(c_i)}$$

Experiments and Evaluation

■ Process of Similarity Test

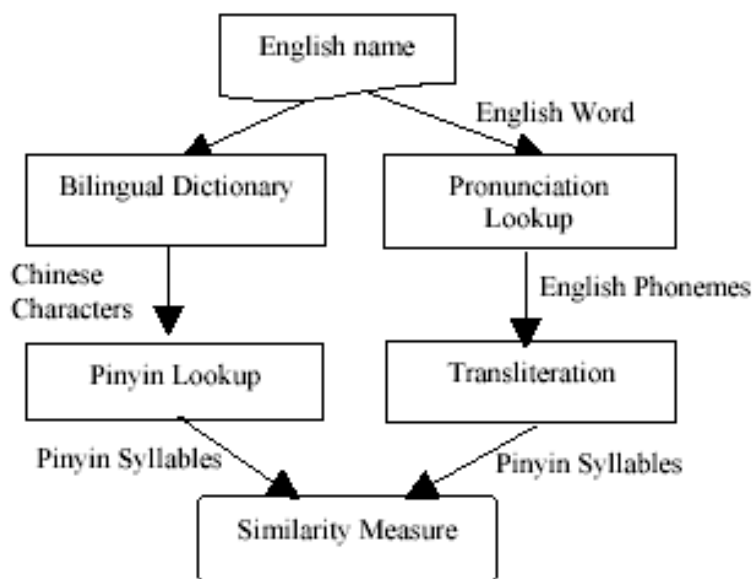


Figure 6. Flowchart of Similarity Test

- LDC's English-Chinese bilingual named entity list 1.0 (beta)
- CMU's pronunciation dictionary
- LDC's Chinese character table with pinyin

- Similarity Measurement: Error rate with edit distance, S1: machine-generated transliteration; S2: standard transliteration

$$e = \frac{d(S_1, S_2)}{L_{S_2}} \quad d(S_1, S_2) = |L_{S_2} - (i + d + s)|$$

Experimental Results

■ Experiment I

■ Configure

Training size for p(C E)	Training size for p(C)	Close test size	Open test size	Top-n used
1500	1500	100	100	1

■ Results

Error rate (%)	0~10	10~30	30~50	>50
Open (100)	7	24	11	58
Close (100)	10	21	35	34
Average (200)	8.5%	22.5%	23%	46%

■ Observations

- In both open and close tests, the possibility of finding the correct transliteration in top-1 result candidates is fairly low: 7% and 10% respectively, and 8.5% in average. (We consider error rate < 10% correct transliteration).
- The close test, 66% (10-34%) machine-generated transliterations are with error rate less than 50%. In open test, 42% (1-58%) are with error rate < 50%;

Experimental Results(cont'd)

- Experiment II

- Configure

Training size for p(C E)	Training size for p(C)	Close test size	Open test size	Top-n used
1500	1500	100	100	20

- Results

Top n	5	10	15	20
Open (100)	14	19	23	29
Close (100)	29	35	51	64
Average (200)	21.5%	27%	37%	46.5%

- Observations

- In average, 46.5% of testing instances can have their correct transliterations (error rate < 10%) with top-20 candidates used.

Experimental Results(cont'd)

■ Experiment III

■ Configure

Training size for p(C E)	Training size for p(C)	Open test size	Top-n used
2233	2233	1541	1

■ Results

Systems	Meng et al.	Virga et al.	Ours
Error rate	52.5%	50.8%	54.2%

■ Observations

- Ours: comparable with other state-of-the-art systems, slight worse;
- In average, the error rates shows that the top-1 transliteration is almost incorrect: how far? 50% difference in edit distance can be nearly two different names, e.g. /ya li shan da/ (Alexander, 亚力山大) and /ya li shi duo de/ (Aristotle, 亚里士多德)

Experimental Results(cont'd)

- Experiment III (cont'd)

Training size for $p(C E)$	Training size for $p(C)$	Test size	Error rate
2233	2233	1541	54.2%
1500	2233	1541	56.5%
2233	1500	1541	67.7%

- Our approach is more sensitive to LM than to transliteration model, thus on the training data on which the LM is trained. Transliteration model only reflects symbol-mapping relationships between sound units instead of the transformation of whole sequences, which is less sensitive to paucity of data.

Discussions and Conclusions

- We haven't taken the context of phonemes into consideration, which would provide important information to disambiguate possible mapping units in pinyin. Although it has not been shown outperform other models, but the advantage is to better accommodate flexible features with respect to dependencies among surrounding phonemes and their mapping units, e.g. under MaxEnt framework.
- Data sparseness can seriously affect LM. Smoothing techniques could be used in future work.

Thank you!