Confusion Network for Arabic Name Disambiguation and Transliteration in Statistical Machine Translation

Young-Suk Lee
IBM T. J. Watson Research Center
1101 Kitchawan Road
Yorktown Heights, NY 10598, USA
ysuklee@us.ibm.com

Abstract

Arabic words are often ambiguous between name and non-name interpretations, frequently leading to incorrect name translations. We present a technique to disambiguate and transliterate names even if name interpretations do not exist or have relatively low probability distributions in the parallel training corpus. The key idea comprises named entity classing at the preprocessing step, decoding of a simple confusion network created from the name class label and the input word at the statistical machine translation step, and transliteration of names at the post-processing step. Human evaluations indicate that the proposed technique leads to a statistically significant translation quality improvement of highly ambiguous evaluation data sets without degrading the translation quality of a data set with very few names.

1 Introduction

Arabic person and location names are often ambiguous between name and non-name interpretations, as noted in (Hermjakob et al., 2008; Zayed et al., 2013). (1) and (2) illustrate such ambiguities for Iraqi Arabic, where the ambiguous names and their translations are in bold-face and the Buckwalter transliteration of Arabic is provided in parentheses:[1]

(1) a. انـي ساكـن بـشقـة يـم الـمـدرسة ب**خضراء**
(Any sAkn b$qp ym Almdrsp b**xDrA'**)
I live in an apartment near the school in **Khadraa**

b. مـصبـوغة **خضراء**
(mSbwgp **xDrA'**)
It is painted **green**

(2) a. شيـقـدر يـقـول إلـك **صبـاح**
($yqdr yqwl Alk **SbAH**)
What can **Sabah** tell you?

b. **صبـاح** الـخيـر إنـت أكـيـد نـقـيـب حسام
(**SbAH** Alxyr Ant Akyd nqyb HsAm)
Good **morning** you must be captain Hosam

In this paper, we propose a technique for disambiguating and transliterating Arabic names in an end-to-end statistical machine translation system. The key idea lies in name classing at the preprocessing step, decoding of a simple confusion network created from the class label *$name*, and the input word at the machine translation step, and transliteration of names by a character-based phrase transliteration model at the post-processing step.

While Bertoldi et al. (2007) propose confusion network decoding to handle multiple speech recognition outputs for phrase translation and Dyer et al. (2008) generalize lattice decoding algorithm to

[1]Arabic should be read from right to left, and the Buckwalter transliteration should be read from left to right.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 433–443, Dublin, Ireland, August 23-29 2014.

tackle word segmentation ambiguities for hierarchical phrase-based translation, the current proposal is the first to deploy a confusion network for name disambiguation and translation. The character-based phrase transliteration model captures the asymmetry between Arabic and English vowel systems by treating English vowels as spontaneous words attachable to the neighboring target phrases for phrase (a sequence of characters) acquisition.

Confusion network decoding enables the system to choose between name and other translations of the source word on the basis of the decoding cost computed from all of the decoder feature functions which incorporate name tag scores into translation model scores. Probabilistic choice between name versus non-name interpretations makes the technique robust to name classing errors, without stipulating the frequency threshold of the names to be transliterated in order to avoid translation quality degradation (Hermjakob et al., 2008; Li et al., 2013). A tight integration of named entity detection and classing into the machine translation system, coupled with a generative approach to name transliteration, enables the system to produce reliable name translations even when name interpretations do not exist or have relatively low distributions in the parallel corpus, distinguishing the current proposal from Hermjakob et al. (2008).

In Section 2, we give an overview of the translation system. In Section 3, we discuss the model training and confusion network decoding. In Section 4, we detail name transliteration model. We present the experimental results in Section 5. We discuss related work in Section 6 and conclude the paper in Section 7.

2 End-to-end Translation System Overview

Arabic name disambiguation and transliteration techniques are incorporated into an end-to-end phrase translation system (Och and Ney, 2002; Koehn et al., 2003; Koehn et al., 2007). Our phrase translation system builds on Tillmann (2003) for translation model training and an in-house implementation of Ney and Tillmann (2003) for beam search phrase decoding.

Iraqi Arabic to English end-to-end phrase translation systems are trained on DARPA TransTac data (Hewavitharana et al., 2013), comprising 766,410 sentence pairs (~6.8 million morpheme tokens in Arabic, ~7.3 million word tokens in English; ~55k unique vocabulary in Arabic and ~35k unique vocabulary in English). The data consist of sub-corpora of several domains including military combined operations, medical, humanitarian aid, disaster relief, etc., and have been created primarily for speech-to-speech translations. The process flow of Arabic to English translation incorporating the proposed technique is shown in Figure 1. The components relevant to name disambiguation and transliteration are in bold face.

Given the input sentence (3), the spelling normalizer normalizes آني to اني.

(3) آني ساكن بشقة يم المدرسة بخضراء
(|ny sAkn b$qp ym Almdrsp bxDrA')

The morpheme segmenter segments a word into morphemes (Lee et al., 2003; Lee, 2004; Habash and Sadat, 2006) as in (4), where # indicates that the morpheme is a prefix.

(4) اني ساكن ب# شقة يم ال# مدرسة ب# خضراء
(Any sAkn b# $qp ym Al# mdrsp b# xDrA')

Part-of-speech tagging is applied to the morphemes, identifying a name with the tag NOUN_PROP. The input word tagged as NOUN_PROP is classified as name, denoted by the label *$name* in (5).

(5) اني ساكن ب# شقة يم ال# مدرسة ب# $name_(خضراء)
(Any sAkn b# $qp ym Al# mdrsp b# $name_(xDrA'))

The token $name_(خضراء) is decomposed into the class label *$name* and the source word بخضراء, creating a simple confusion network for decoding. The beam search phrase decoder computes the translation costs for all possible input phrases including the phrase pair "*$name* | *$name*",[2] using all of

[2] The source phrase *$name* translates to the target phrase *$name*.

the decoder feature functions. Assuming that the translation cost for *$name* being translated into *$name* is the lowest, the decoder produces the translation (6), where the name classed source word خضراء retains its Arabic spelling .

(6) I live in an apartment near the school in خضراء

The Arabic word خضراء in (6) is transliterated into *khadraa* by the NAME/OOV transliteration module. And the system produces the final translation output (7).

(7) I live in an apartment near the school in khadraa

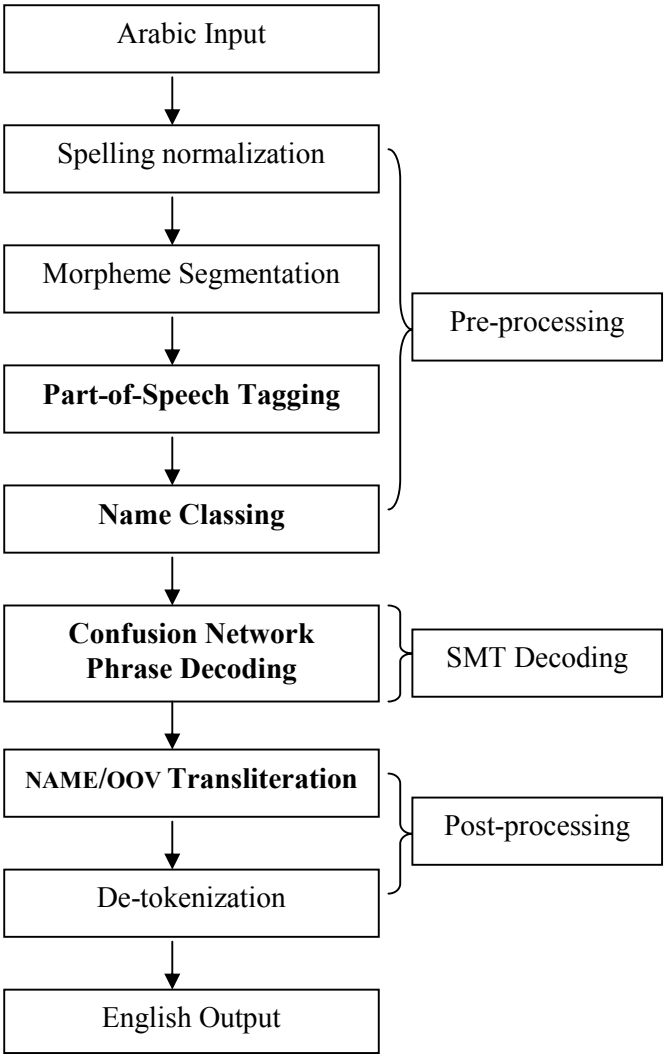

Figure 1. Process Flow of Arabic to English Phrase Translation Decoding

We use an in-house implementation of the maximum entropy part-of-speech tagger described in Adwait (1996) for name classing. The part-of-speech tagger is trained on the combination of LDC-released Arabic Treebank data containing about 3 million morpheme tokens from MSA (modern standard Arabic) and in-house annotated TransTac Iraqi Arabic data containing about 63k morpheme tokens.

F-score of the tagger on proper noun tags, NOUN_PROP, is about 93% on 2,044 MSA name tokens derived from Arabic Treebank: Part 3 v 3.2 (LDC2010T08), and about 81.4% on 2,631 Iraqi Arabic name tokens derived from the DARPA TransTac corpus.

3 Model Training and Confusion Network Decoding

We train translation and language models with name classing to obtain proper translation and language model probabilities of the class label *$name*. We extend the baseline phrase beam search decoder to handle a relatively simple confusion network (CN hereafter) and incorporate the name part-of-speech tagging scores into the decoder feature functions.

3.1 Translation Model

For any name classed input word, *$name_(orig)* in (5), we would like to have the name translation, *$name* → *$name*, always available in addition to other translations of the input word obtainable from the parallel training corpus.

In order to estimate *$name* distributions without obfuscating the distributions of other training vocabulary, we apply name classing only to words that occur less than 3 times in the training corpus and part-of-speech tagged with NOUN_PROP. The reasons are three-fold: 1) we need to keep all non-name translations of the training vocabulary, 2) typical low frequency words include names and typos, 3) even with *$name* classing on low frequency words only, the overall *$name* count is high enough for a robust probability estimation.

After name classing of words occurring less than 3 times, *$name* occurs 6,944 times (122th most frequent token) in Arabic and 9,707 times (108th most frequent token) in English. We train both phrase translation and distortion models on the name classed parallel corpus. Note that the frequency restriction applies only to model training. During decoding, any word labeled with *$name* may be name transliterated regardless of its frequency in the training corpus, differentiating the current technique from (Li et al., 2013).

3.2 Language Models

To properly capture the name and non-name ambiguities, we interpolate two types language models: 1) 5-gram language model trained on the English side of the parallel corpus without name classing (LM1), 2) 5-gram language model trained on the English side of the parallel corpus and additional monolingual corpora with name classing (LM2).

Each language model is smoothed with modified Kneser-Ney (Chen and Goodman, 1998). The two sets of language models are interpolated, as in (8), where α is set to 0.1. We find the optimal interpolation weight on the basis of BLEU scores of the development test data set containing about 30k word tokens in Arabic and about 43k word tokens in English.

(8) $\alpha \cdot LM1 + (1-\alpha) \cdot LM2$

3.3 Confusion Network Decoding

The confusion network containing the class label *$name* and the source word is handled by an extension of the baseline phrase decoder. The baseline decoder utilizes 11 feature functions including those in (9)[3] through (14), where $\bar{f}$ denotes the source phrase and $\bar{e}$, the target phrase, and ***s***, the source sentence, ***t***, the target sentence and *a*, a word alignment. We use the in-house implementation of the simplex algorithm in Zhao et al. (2009) for decoder parameter optimization.

(9) Direct phrase translation model for $pr(\bar{e} \mid \bar{f})$
(10) Distortion models(Al-Onaizan and Papineni, 2006)
(11) Mixture language models

[3] We do not use $pr(\bar{f} \mid \bar{e})$

(12) Lexical weights $p_w(\bar{f}|\bar{e},a)$ & $p_w(\bar{e}|\bar{f},a)$, cf. (Koehn et al., 2003)
(13) Lexical weights p_w (**t**|**s**,a) & p_w (**s**|**t**,a)
(14) Word and phrase penalties (Zens and Ney, 2004)

Lexical weight $p_w(\bar{e}|\bar{f},a)$ in (12) is computed according to (15), where $j = 1, \ldots, n$ source word positions and $i = 1,\ldots, m$ target word positions within a phrase, N = source phrase length, $w(e|f)$ = the lexical probability distribution:[4]

(15) $$p_w(\bar{e}|\bar{f},a) = (\prod_{j=1}^{n} w_{\max(j,i)\in a}(e_i|f_j)) / N$$

Lexical weight $p_w(\mathbf{t}|\mathbf{s},a)$ in (13) is computed according to (16), where K = number of phrases in the input sentence, $k = k^{th}$ phrase, and $pr_{w\_k}(\bar{e}|\bar{f},a) = p_{w\_k}(\bar{e}|\bar{f},a)$ without normalization by the source phrase length N.

(16) $$p_w(\mathbf{t}|\mathbf{s},a) = \prod_{k=1}^{K} pr_{w\_k}(\bar{e}|\bar{f},a)$$

We augment the baseline decoder in two ways: First, we incorporate the maximum entropy part-of-speech tagging scores of names into the translation scores in (9), (12) and (13). We simply add the name part-of-speech tag cost, i.e. –log probability, to the translation model costs. Second, the decoder can activate more than one edge from one source word position to another, as shown in Figure 2.[5] The name classed input is split into two tokens *$name* and *xDrA'*, leading to two separate decoding paths. The choice between the two paths depends on the overall decoding cost of each path, computed from all of the decoder feature functions.

Since the decoding path to *$name* is always available when the input word is classed as *$name* at the pre-processing step, the technique can discover the name interpretation of an input word even if the name interpretation is absent in the parallel training corpus. Even when the input word occurs as a name in the training corpus but has a lower name translation probability than non-name translations in the baseline phrase table, it can be correctly translated into a name as long as the word is labeled as *$name* and the decoder feature functions support the *$name* path in the given context. When a non-name token is mistakenly labeled as *$name*, the confusion network decoder can recover from the mistake if the non-name path receives a lower decoding cost than the *$name* path.[6] If the input token is name classed and the correct name translation also exists in the baseline phrase table with a high probability, either path will lead to the correct translation, and the decoder chooses the path with the lower translation cost.

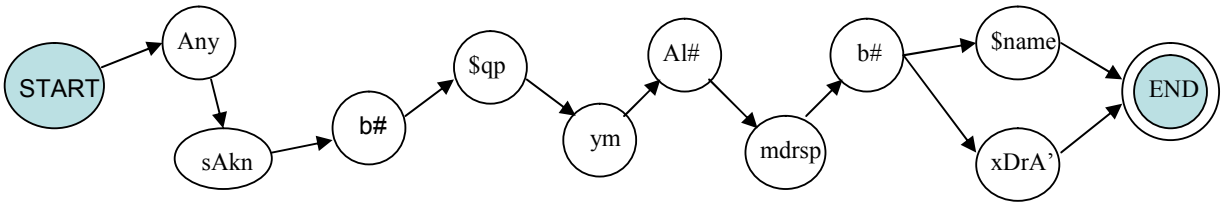

Figure 2. Confusion Network Decoding Paths for Name Classed Input

[4] Estimated in the manner described in Koehn et al. (2003).
[5] Arabic is represented by Buckwalter transliteration scheme.
[6] The decoding scores are computed as cost on the basis of –log likelihood of various component models. And therefore, a smaller decoding cost indicates a higher translation quality.

4 Character-Based Phrase Transliteration Models

All instances of un-translated input words, which include names and OOVs, are transliterated in the post-processing step. Character-based phrase transliteration models are trained on 9,737 unique name pairs. 965 name pairs are obtained from a name lexicon and the remaining 8,772 name pairs are automatically derived from the parallel training corpus as follows: 1) Take each side of the parallel corpus, i.e. Iraqi Arabic or English. 2) Mark names manually or automatically. 3) Apply word alignment to the name-marked parallel corpus in both directions. 4) Extract name pairs aligned in both directions. For name marking, we used the manual mark-up that was provided in the original data.

5-gram character language models are trained on about 120k entries of names in English. In addition to about 9.7k names from the English side of the parallel names, about 110k entries are collected from wiki pages, English Gigaword 5th Edition (LDC2011T07), and various name lexicons.

4.1 Phrase Extraction with English Vowels as Spontaneous Words

Short vowels are optional in written Arabic, whereas all vowels have to be obligatorily specified in English for a word to be valid (Stalls and Knight, 1998; Al-Onaizan and Knight, 2002b). We model the asymmetrical nature of vowels between the two languages by treating all instances of unaligned English vowels – *a, e, i, o, u* – as spontaneous words which can be attached to the left or to the right of an aligned English character for phrase extractions. An example GIZA++ (Och and Ney, 2003) character alignment is shown in Figure 3. Arabic name is written left to right to illustrate the monotonicity of the alignments.

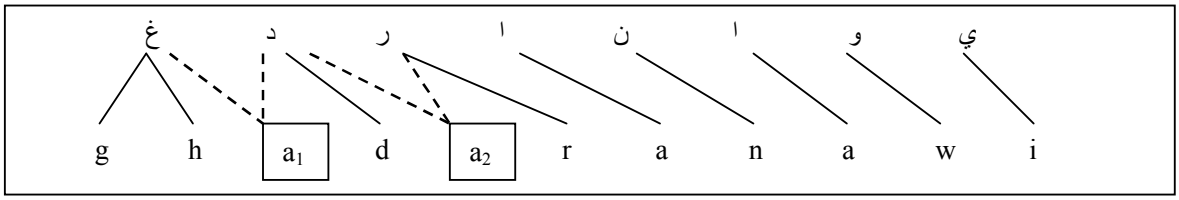

Figure 3. Automatic Character Alignment between Arabic and English names

In Figure 3, solid lines indicate the automatic machine alignments. English vowels in rectangular boxes indicate null alignments by the aligner. The dotted lines indicate the potential attachment sites of the unaligned vowels for phrase extractions. The first instance of unaligned *a* (denoted by a_1) may be a part of the phrases containing the preceding consonant sequence *g h*, or the following consonant *d*. The second instance of unaligned *a* (denoted by a_2) may be a part of the phrases containing the preceding consonant *d* or the following consonant *r*.[7]

4.2 Experiments

We use exact match accuracy[8] to evaluate transliteration qualities. Systems are tested on 500 unique name pairs including 52 names unseen in the training corpus. Experimental results are shown in Table 1.[9] Note that using English vowels as spontaneous words dramatically improves the accuracy from 21.6% to 89.2%.

Decoding is carried out by the baseline phrase decoder discussed in Section 3.3, using the same decoder feature functions except for the distortion models. Using only phrase translation and language model probabilities for decoding results in 74.4% accuracy on SYSTEM4, much lower than 90% accuracy with all decoder feature functions. The same language model is used for all experiments. For the end-to-

[7] Attachment of unaligned English vowels takes place after phrase extractions and should be distinguished from a heuristic alignment of unaligned English vowels to Arabic characters before phrase extractions.
[8] A transliteration is correct if and only if it exactly matches the truth, i.e. gold standard.
[9] GIZA++ word aligner is trained with 5 iterations of IBM MODEL 1, 5 iterations of HMM, 5 iterations of IBM MODEL 3 and 5 iterations of IBM MODEL 4. HMM word aligner (Vogel et al., 1996) is trained with 15 iterations of IBM MODEL 1 and 6 iterations of HMM.

end translation quality evaluations in Section 5, we use SYSTEM4. Exact match accuracy of SYSTEM4 on the 52 unseen name pairs is 46%.

Systems	Character Alignments	Symmetrization[10]	Target spontaneous words	Accuracy
SYSTEM1	GIZA++	Union	None	21.6%
SYSTEM2	HMM	Refined	None	86.8%
SYSTEM3	GIZA++	Union	All English vowels: *a, e, i, o, u*	89.2%
SYSTEM4	GIZA++ & HMM	Union	All English vowels: *a, e, i, o, u*	90.0%

Table 1. Name transliteration accuracy on 500 names according to various phrase extraction techniques

5 End-to-end Translation System Experimental Results

End-to-end translation quality experiments are carried out on 3 evaluation data sets shown in Table 2. TransTac.eval has a low out-of-vocabulary (OOV) and a low name ratios, and has been used as the test data for system development among DARPA BOLT–C[11] program partners. TransTac.oov has a high OOV and a high name ratios, and has been created in-house for OOV detection system developement. TransTac.name has a low OOV and a high name ratios, and was used for the TransTac 2008 name translation evaluations.

Evaluation Data Sets	TransTac.eval	TransTac.oov	TransTac.name
sentence count	3,138	344	79
token count	36,895	3,053	514
OOV ratio	0.4%	4.7%	0.6%
name ratio	~0.5%	~11.3%	~15.4%

Table 2. Translation Quality Evaluation Data Statistics

5.1 Systems, Metrics and Results

End-to-end translation system evaluation results are shown in Table 3. Bold-faced and italicized scores indicate that the system's translation quality is statistically significantly better than all other systems with over 95% confidence, i.e. two-tailed P value < 0.05 in paired t-tests.

Metrics	EvalSets / Systems	TransTac.eval	TransTac.oov	TransTac.name	TransTac.name_spnorm
Uncased BLEU (4-gram & 1 ref)	baseline	33.35	30.72	35.03	37.39
	OOVTranslit	33.35	31.93	35.03	37.54
	name_t	32.94	31.81	32.97	40.15
	CN	33.35	32.60	32.19	40.97
HUMAN (6-point scale)	baseline	3.16	1.45	3.19	3.19
	OOVTranslit	3.22	2.88	3.36	3.36
	name_t	2.16	2.79	3.58	3.58
	CN	3.20	***3.09***	***3.86***	***3.86***

Table 3. Translation Quality Evaluation Result

The system *baseline* is trained without name classing and decoded by the baseline decoder without name classing. The system OOVTranslit is trained and decoded the same way as the baseline except that all instances of un-translated OOVs are transliterated at the post-processing step. The system *name_t* is

[10] Bi-directional word alignment symmetrization methods, as defined in Och and Ney (2003), include union, intersection and refined.

[11] BOLT stands for Broad Operational Language Translation and BOLT-C focuses on speech-to-speech translation with dialog management.

trained without name classing and decoded by the baseline decoder with name classing.[12] The system *CN* is trained with name classing and decoded by the CN decoder with name classing.[13]

We evaluate the systems, using automatic BLEU (Papineni et al., 2002), and 6-point scale human evaluations. Lowercased BLEU scores are computed with 1 reference translation up to 4-grams. Scoring criteria for human evaluations are as follows. **0**: exceptionally poor; **1**: poor; **2**: not good enough; **3**: good enough; **4**: very good; **5**: excellent. Human evaluations are conducted on a subset of the automatic evaluation data containing names.[14] We exclude the input sentences for which all systems produce the same translation output. This leaves 201 sentences from TransTac.eval, 197 sentences from TransTac.oov, 64 sentences from TransTac.name.

5.2 Result Analysis

We observe that human evaluation scores are relatively consistent with BLEU scores on two data sets, TransTac.eval and TransTac.oov. TransTac.eval contains very few names. Therefore, incorrect name classing at the pre-processing step hurts the translation quality for the system *name_t*. The CN decoder can improve the translation quality by recovering from a name classing error by choosing the non-name path. Transliteration of OOVs (OOVTranslit) can improve the translation quality if any of the OOVs are names. Human evaluations capture the behaviors of the CN decoder and OOVTranslit by giving a slightly higher (statistically insignificant) score to OOVTranslit, 3.22, and the CN decoder, 3.20, than to the baseline, 3.16. All three systems, baseline, OOVTranslit and CN, however, received the same BLEU scores, 33.35. This seems to reflect the fact humans can easily capture the spelling variation of names whereas the automatic evaluation with 1 reference cannot.

Transtac.oov has a high OOV and a high name ratios and all OOVs are names. Therefore, name classing improves the translation quality as long as the correctly classed names out-number the incorrectly classed ones, explaining the higher translation quality of *name_t* than the baseline. OOVTranslit improves the translation quality over the baseline because all OOVs are names. The CN decoder out-performs all three other systems by correctly disambiguating non-OOV names and transliterating name OOVs. BLEU scores and human evaluation scores show the same pattern.

For TransTac.name with a high name and a low OOV ratios, however, human evaluation and BLEU scores show the opposite pattern, although none of the BLEU scores are statistically significantly better than others (note the small evaluation data size of 79 segments and 514 tokens). Since most names in this data set are known to the translation vocabulary and is highly ambiguous, we expect the CN decoder to out-perform all other systems. This expectation is borne out in the human evaluations, but not in BLEU scores. Our analysis indicates that the apparent inconsistency between BLEU and human evaluation scores is primarily due to spelling variations of a name, which are not captured by BLEU with just one reference, cf. (Li et al., 2013). Out of the human evaluated 64 names in TransTac.name, the baseline system produced the same spelling as the reference 34 times (53.13%), which contrasts with 28 times (43.75%) by the CN decoder. Overall, the CN decoder produced 62 correct name translations, about 20% more than 49 correct translations by the baseline system. Table 4 shows the names for which the reference spelling agrees with the baseline system, but disagrees with the CN decoding followed by transliteration.

Reference	CN output	Reference	CN output	Reference	CN output
tikrit	*tikreet*	*mariam*	*maryam*	*mousa*	*moussa*
ajlan	*al-`ajlan*	*jaafar*	*gaafar*	*basra*	*al-basra*

Table 4. Name Spelling Variations

[12] We ensure that any name classed input word *$name* is translated into *$name* by adding *$name* to the translation vocabulary, and the input word for *$name* is transliterated in the post-processing stage.

[13] We also evaluated another system, called name_st, which is trained with name classing and decoded with name classing using the baseline decoder. BLEU scores on TransTac.eval and TransTac.oov indicated that model training and decoding with name classing (name_st) is only slightly better than model training without name classing and decoding with name classing (name_t).

[14] For TransTac.eval data, we selected the sentences containing words tagged as name, i.e. NOUN_PROP, by the automatic part-of-speech tagger. The name ratio around 0.5% in Table 2 is computed on the basis of human annotations on the reference translation.

To verify that the inconsistency between BLEU and human evaluation scores is due to name spelling variations which humans capture but automatic metrics does not, we recomputed BLEU scores after normalizing spellings of the system outputs to be consistent with the reference translation spelling. The recomputed BLEU scores are denoted by TransTac.name_spnorm in Table 3, which shows that the recomputed BLEU scores are indeed consistent with the human evaluation scores.[15] Also note that the translation quality improvement by transliterating OOV names is well captured in human evaluation scores, 3.19 in the baseline vs. 3.36 in the system OOVTranslit, but not in BLEU scores, 35.03 in both baseline and OOV-Translit.

We point out that the same name is often spelled differently in various parts of our training corpus and even in the same reference translation, e.g. *al-aswad* vs. *aswad*, *jassim* vs. *jasim*, *risha* vs. *rasha*, *mahadi* vs. *mehdi* vs. *mahdi*, etc., as had been noted in Al-Onaizan and Knight (2002b), Huang et al. (2008).

6 Related Work

Al-Onaizan and Knight (2002a) propose an Arabic named entity translation algorithm that performs at near human translation accuracy when evaluated as an independent name translation module. Hassan et al. (2007) propose to improve named entity translation by exploiting comparable and parallel corpora. Hermjakob et al. (2008) present a method to learn when to transliterate Arabic names. They search for name translation candidates in large lists of English words/phrases. Therefore, they cannot accurately translate a name if the correct English name is missing in the word lists. Their restriction of named entity transliteration to rare words cannot capture name interpretations of frequent words, e.g. صباح (Sabah/morning), if the name interpretations are absent in the parallel corpus. Li et al. (2013) propose a Name-aware machine translation approach which tightly integrates high accuracy name processing into a Chinese-English MT model. Similar to Hermjakob et al. (2008), they restrict the use of name translation to names occurring less than 5 times in the training data. They train the translation model by merging the name-replaced parallel data with the original parallel data to prevent the quality degradation of high frequency names.

Onish et al. (2010) present a lattice decoding for paraphrase translations, which can handle OOV phrases as long as their paraphrases are found in the training corpus. They build the paraphrase lattices of the input sentence, which are given to the Moses lattice decoder. They deploy the source-side language model of paraphrases as a decoding feature.

Stalls and Knight (1998) propose a back-transliteration technique to recover original spelling in Roman script given a foreign name or a loanword in Arabic text, which consist of three models: a model to convert an Arabic string to English phone sequences, a model to convert English phone sequences to English phrases, a language model to rescore the English phrases. They use weighted finite state transducers for decoding. Al-Onaizan and Knight (2002b) propose a spelling-based source-channel model for transliteration (Brown et al., 1993), which directly maps English letter sequences into Arabic letter sequences, and therefore overcomes Stalls and Knight's major drawback that needs a manual lexicon of English pronunciations. Sherif and Kondrak (2007) propose a substring-based transliteration technique inspired by phrase based translation models and show that substring (i.e. phrase) models out-perform letter (i.e. word) models of Al-Onaizan and Knight (2002b). Their approach is most similar to the current approach in that we both adopt phrase-based translation models for transliteration. The current approach and Sherif and Kondrak (2007), however, diverge in most technical details including word alignments, phrase extraction heuristics and decoding, although it is not clear how they estimate transliteration probabilities. Crucially, we use the same set of decoder feature functions (excluding distortion models) as the end-to-end phrase translation system including lexical weights for phrases and a sentence in both directions and word/phrase penalties, whereas Sherif and Kondrak (2007) use only transliteration and language models for substring

[15] The spellings of the CN decoder output are normalized as follows: 38 instances of names, 2 instances of *'s* to *is*, 2 instances of *the city of arar* to *arar city* and 1 instance of *talk with* to *speak to*. Only name spelling normalizations were necessary for other system outputs.

transducer. We noted in Section 4 that inclusion of all decoder feature functions improves the accuracy by 15.6% absolute, compared with using just translation and language models for decoding.

7 Conclusion

We proposed a confusion network decoding to disambiguate Arabic names between name and non-name interpretations of an input word and character-based phrase transliteration models for NAME/OOV transliteration.

Name classing at the pre-processing step, coupled with name transliteration at the post-processing step, enables the system to accurately translate OOV names. Robust TM/LM probability estimations of names on the class label *$name* enable the system to correctly translate names even when the name interpretation of an in-vocabulary word is absent from the training data. Confusion network decoding can recover from name classing errors by choosing an alternative decoding path supported by decoder feature functions, obviating the need for stipulating a count threshold of an input token for name translation. The character-based phrase transliteration system achieves 90% exact match accuracy on 500 unique name pairs, utilizing all of the phrase decoder feature functions except for distortion models. We capture the asymmetries of English and Arabic vowel systems by treating any instance of an unaligned English vowel as a spontaneous word that can be attached to the preceding or following target phrases for phrase acquisition.

Although we proposed the confusion network decoding and character-based phrase transliteration models in the contexts of Arabic name disambiguation and transliteration tasks, the techniques are language independent and may be applied to any languages.

Acknowledgements

This work has been funded by the Defense Advanced Research Projects Agency BOLT program, Contract No. HR0011-12-C-0015. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the view of DARPA. We would like to thank Lazkin Tahir for his tireless effort on human evaluations. We also thank anonymous reviewers for their helpful comments and suggestions.

References

Y. Al-Onaizan and K. Knight. 2002. Translating Named Entities Using Monolingual and Bilingual Resources. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 400–408.

Y. Al-Onaizan and K. Knight. 2002. Machine Transliteration of Names in Arabic Text. In *Proceedings of the Association for Computational Linguistics Workshop on Computational Approaches to Semitic Languages*.

Y. Al-Onaizan and K. Papineni. 2006. Distortion models for Statistical Machine Translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 529–536.

N. Bertoldi, R. Zens, and M. Federico.2007. Speech translation by confusion network decoding. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1297–1300.

P. Brown, S. Della Pietra, V. Della Pietra and R. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. In *Computational Linguistics, 19(2)*, pages 263–311.

S. Chen and J. Goodman. 1998. An Empirical Study of Smoothing Techniques for Language Modeling. TR-10-98. Computer Science Group. Harvard University.

C. Dyer, S. Muresan, and P. Resnik. 2008. Generalizing Word Lattice Translation. In *Proceeding of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1012–1020.

N. Habash and F. Sadat. 2006. Arabic Preprocessing Schemes for Statistical Machine Translation, In *Proceedings of the North American Chapter of the Association for Computational Linguistics* (NAACL), pages 49–52.

A. Hassan, H. Fahmy, and H. Hassan. 2007. Improving Named Entity Translation by Exploiting Comparable and Parallel Corpora. In *Proceeding RANLP'07*, pages 1–6.

U. Hermjakob, K. Knight, and H. Daume III. 2008. Name Translation in Statistical Machine Translation:Learning When to Transliterate. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 389–397.

S. Hewavitharana, D. Mehay, S. Ananthakrishnan, and P. Natarajan. 2013. Incremental Topic-Based Translation Model Adaptation for Conversational Spoken Language Translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 697-701.

F. Huang, A. Emami, and I. Zitouni. 2008. When Harry Met Harri, هاری and 亨利 : Cross-lingual Name Spelling Normalization. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 391–399.

P. Koehn, F. Josef Och, and D. Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology – Volume 1*, pages 127–133.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics on Interactive Poster and Demonstration Sessions*, pages 177–180.

Y. Lee, K. Papineni, S. Roukos, O. Emam and H. Hassan. 2003. In *Proceedings of the 41st Annual Meeting of Association for Computational Linguistics – Volume 1*, pages 399–406.

Y. Lee. 2004. Morphological Analysis for Statistical Machine Translation. In *Proceedings of Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics: Short Papers*, pages 57–60.

H. Li, J. Zheng, H. Ji, Q. Li and W. Wang. 2013. Name-aware Machine Translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 604–614.

F. Och and H. Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. In *Computational Linguistics 29(1)*, pages 19–51. MIT Press.

T. Onish, M. Utiyama and E. Sumita. 2010. Paraphrase Lattice for Statistical Machine Translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics Short Papers*, pages 1–5.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

A. Ratnaparkhi. 1996. A Maximum Entropy Model for Part-Of-Speech Tagging. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 133–142.

T. Sherif and G. Kondrak. 2007. Substring-Based Transliteration. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics,* pages 944–951.

B. G. Stalls and K. Knight. 1998. Translating Names and Technical Terms in Arabic Text. *In Proceedings of the COLING/ACL Workshop on Computational Approaches to Semitic Languages*.

C. Tillmann. 2003. A Projection Extension Algorithm for Statistical Machine Translation. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 1–8.

C. Tillmann and H. Ney. 2003. Word Reordering and a Dynamic Programming Beam-Search Algorithm for Statistical MT. *Computational Linguistics 29(1)*, pages 97–133. MIT Press.

S. Vogel, H. Ney and C. Tillmann. 1996. HMM-based word alignment in statistical machine translation. In *Proceedings of the 16th International Conference on Computational Linguistics*, Volume 2, pages 836–841.

O. Zayed, S. El-Beltagy, O. Haggag. An Approach for Extracting and Disambiguating Arabic Person's Names Using Clustered Dictionaries and Scored Patterns. In *Natural Language Processing and Information Systems Lecture Notes in Computer Science*. Vol. 7934, 2013, pages 201–212.

B. Zhao and S. Chen. 2009. A Simplex Armijo Downhill Algorithm for Optimizing Statistical Machine Translation Decoding. In *Proceedings of Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics Short Papers*, pages 21–24.

R. Zen and H. Ney. 2004. Improvements in phrase-based statistical machine translation. In *Proceedings of Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics*, pages 257–264.