Language Portability for Dialogue Systems: Translating a Question-Answering System from English into Tamil

Satheesh Ravi[1] and **Ron Artstein**[1,2]
University of Southern California
[1]Department of Computer Science, 941 Bloom Walk, Los Angeles, CA 90089-0781, USA
[2]Institute for Creative Technologies, 12015 Waterfront Drive, Playa Vista CA 90094, USA
satheesr@usc.edu, artstein@ict.usc.edu

Abstract

A training and test set for a dialogue system in the form of linked questions and responses is translated from English into Tamil. Accuracy of identifying an appropriate response in Tamil is 79%, compared to the English accuracy of 89%, suggesting that translation can be useful to start up a dialogue system. Machine translation of Tamil inputs into English also results in 79% accuracy. However, machine translation of the English training data into Tamil results in a drop in accuracy to 54% when tested on manually authored Tamil, indicating that there is still a large gap before machine translated dialogue systems can interact with human users.

1 Introduction

Much of the effort in creating a dialogue system is devoted to the collection of training data, to allow the system to process, understand, and react to input coming from real users. If a comparable system is available for a different language, it would be helpful to use some of the existing foreign language resources in order to cut down the development time and cost – an issue known as *language portability*. Recent efforts have shown machine translation to be an effective tool for porting dialogue system resources from French into Italian (Jabaian et al., 2010; Jabaian et al., 2013; Servan et al., 2010); this system used concept-based language understanding, and the findings were that machine translation of the target language inputs yielded better results than using translation to train an understanding component directly for the target language. Here we report similar findings on more challenging data, by exploring a dialogue system with a less structured understanding component, using off-the-shelf rather than domain-adapted machine translation, and with languages that are not as closely related.

Question-answering characters are designed to sustain a conversation driven primarily by the user asking questions. Leuski et al. (2006) developed algorithms for training such characters using linked questions and responses in the form of unstructured natural language text. Given a novel user question, the character finds an appropriate response from a list of available responses, and when a direct answer is not available, the character selects an "off-topic" response according to a set policy, ensuring that the conversation remains coherent even with a finite number of responses. The response selection algorithms are language-independent, also allowing the questions and responses to be in separate languages. These algorithms have been incorporated into a tool (Leuski and Traum, 2011) which has been used to create characters for a variety of applications (e.g. Leuski et al., 2006; Artstein et al., 2009; Swartout et al., 2010). To date, most characters created using these principles understood and spoke only English; one fairly limited character spoke Pashto, a language of Afghanistan (Aggarwal et al., 2011).

To test language portability we chose Tamil, a Dravidian language spoken primarily in southern India. Tamil has close to 70 million speakers worldwide (Lewis et al., 2015), is the official language of Tamil Nadu and Puducherry in India (Wasey, 2014), and an official language in Sri Lanka and Singapore. There is active development of language processing tools in Tamil such as stemmers (Thangarasu and Manavalan, 2013), POS taggers (Pandian and Geetha, 2008), constituent and dependency parsers (Saravanan et al., 2003; Ramasamy and Žabokrtský, 2011), sentence generators (Pandian and Geetha, 2009), etc.; commercial systems are also available, such as

Proceedings of the SIGDIAL 2016 Conference, pages 111–116,
Los Angeles, USA, 13-15 September 2016. ©2016 Association for Computational Linguistics

Google Translate[1] between Tamil and English. Information-providing spoken dialogue systems have been developed for Tamil (Janarthanam et al., 2007), but we are not aware of any conversational dialogue systems.

The main questions we want to answer in this paper are: (Q1) How good is a dialogue system created using translation between English and Tamil? (Q2) Is there a difference between manual and machine translation in this regard? (Q3) Can machine translation be used for interaction with users, that is with manually translated test data?

To answer these questions, we translated linked questions and responses from an English question-answering system into Tamil both mechanically and manually, and tested the response selection algorithms on the English and both versions of the Tamil data. We found that translation caused a drop in performance of about 10% on either manually or mechanically translated text, answering a tentative *fair* to Q1 and *no* to Q2. The answer to Q3 is mixed: a similar performance drop of about 10% was found with machine translation on the target language inputs (that is, translating test questions from Tamil into English); a much more severe drop in performance was observed when using machine translation to create a system in the target language (that is, translating the training data from English into Tamil, and testing on manually authored Tamil). The remainder of the paper describes the experiment and results, and concludes with directions for future research.

2 Method

2.1 Materials

Our English data come from the *New Dimensions in Testimony* system, which allows people to ask questions in conversation with a representation of Holocaust Survivor Pinchas Gutter; this system had undergone an extensive process of user testing, so the linked questions and responses contain many actual user questions that are relevant to the domain (Artstein et al., 2015; Traum et al., 2015). The *New Dimensions in Testimony* system has more than 1700 responses, almost 7000 training questions, and 400 test questions, with a many-to-many linking between questions and responses (Traum et al., 2015). To get a dataset that is large enough to yield meaningful results yet small enough to translate manually, we needed a subset that included questions with multiple links, and answers that were fairly short. We selected all the test questions that had exactly 4 linked responses, and removed all the responses that were more than 200 characters in length; this yielded a test set with 28 questions, 45 responses, and 63 links, with each test question linked to between 1 and 4 responses. We took all the training questions linked to the 45 test responses, resulting in a training set with 441 questions and 1101 links. This sampling procedure was deliberately intended to enable high performance on the English data, in order to provide a wide range of possible performance for the various translated versions.

Automatic translation into Tamil was done using Google Translate, and manual translation was performed by the first author. Thus, each question and response in the training and test datasets has three versions: the original English, and automatic and manual translations into Tamil.

2.2 Tokenization

We use unigrams as tokens for the response classification algorithm; these are expected to work well for Tamil, which has a fairly free word order (Lehmann, 1989). The English text was tokenized using the `word_tokenize` routine from NLTK (Bird et al., 2009). This tokenizer does not work for Tamil characters, so we used a simple tokenizer that separates tokens by whitespace and removes periods, exclamation marks, question marks and quotation marks. The same simple tokenizer was used as a second option for the English text.

2.3 Stemming

Tamil is an agglutinative language where stems can take many affixes (Lehmann, 1989), so we experimented with a stemmer (Rajalingam, 2013).[2] For comparison, we also ran the English experiments with the `SnowballStemmer("english")` routine from NLTK.[3]

2.4 Response ranking

We reimplemented parts of the response ranking algorithms of Leuski et al. (2006), including both the language modeling (LM) and cross-language modeling (CLM) approaches. The LM approach

[1] http://translate.google.com

[2] https://github.com/rdamodharan/tamil-stemmer

[3] http://www.nltk.org/howto/stem.html

constructs language models for both questions and responses using the question vocabulary. For each training question S, a language model is estimated as the frequency distribution of tokens in S, smoothed by the distribution of tokens in the entire question corpus (eq. 1). The language model of a novel question Q is estimated as the probability of each token in the vocabulary coinciding with Q (eq. 2). Each available response R is associated with a pseudo-question Q_R made up by the concatenation of all the questions linked to R in the training data. The responses are ranked by the distance between a novel question Q and the associated pseudo-questions Q_R using Kullback-Leibler (KL) divergence (eq. 3).

$$(1) \quad \pi_S(w) = \lambda_\pi \frac{\#_S(w)}{|S|} + (1-\lambda_\pi)\frac{\sum_{S'} \#_{S'}(w)}{\sum_{S'} |S'|}$$

$$(2) \quad P(w|Q) \cong \frac{\sum_{S'} \pi_{S'}(w) \prod_{q \in \text{tok}(Q)} \pi_{S'}(q)}{\sum_{S'} \prod_{q \in \text{tok}(Q)} \pi_{S'}(q)}$$

$$(3) \quad D(Q||Q_R) = \sum_{w \in V_{S'}} P(w|Q) \log \frac{P(w|Q)}{\pi_{Q_R}(w)}$$

In eq. (1), $\#_S(w)$ is the number of times token w appears in sequence S; $|S|$ is the length of sequence S; the variable S' iterates over all the questions in the corpus, and λ_π is a smoothing parameter. The sum in eq. (2) is over all the questions in the training corpus; the product iterates over the tokens in the question, and thus is an estimate the probability of the question Q given a training question S'. In eq. (3), $V_{S'}$ is the entire question vocabulary.

The CLM approach constructs language models for both questions and responses using the response vocabulary. The language model of a response is estimated in a similar way to eq. (1), but with the smoothing factor using the response corpus (eq. 4). The language model associated with a novel question Q represents the ideal response to Q, and is estimated as the probability of each token in the response vocabulary coinciding with Q in the linked question-response training data (eq. 5); available responses are ranked against this ideal response (eq. 6).

$$(4) \quad \phi_R(w) = \lambda_\phi \frac{\#_R(w)}{|R|} + (1-\lambda_\phi)\frac{\sum_{R'} \#_{R'}(w)}{\sum_{R'} |R'|}$$

$$(5) \quad P(w|Q) \cong \frac{\sum_j \phi_{R_j}(w) \prod_{q \in \text{tok}(Q)} \pi_{S_j}(q)}{\sum_j \prod_{q \in \text{tok}(Q)} \pi_{S_j}(q)}$$

$$(6) \quad D(Q||R) = \sum_{w \in V_{R'}} P(w|Q) \log \frac{P(w|Q)}{\phi_R(w)}$$

The sum in eq. (5) is over all linked question-response pairs $\{S_j, R_j\}$ in the training data, and the product is an estimate the probability of the question Q given the training question S_j. In eq. (6), $V_{R'}$ is the entire response vocabulary.

We did not implement the parameter learning of Leuski et al. (2006); instead we use a constant smoothing parameter $\lambda_\pi = \lambda_\phi = 0.1$. We also do not use the response threshold parameter, which Leuski et al. (2006) use to determine whether the top-ranked response is good enough. Instead, we just check for the correctness of the top-ranked response.

2.5 Procedure

Our basic tests kept the language and processing options the same for questions and responses. Each dataset (English and the two Tamil translations) was processed with both the LM and CLM approaches, both with and without a stemmer; English was also processed with the two tokenizer options.

Additionally, we processed some cross-language datasets, with questions in Tamil and responses in English, and vice versa. We also performed two tests intended to check whether it is feasible to use machine-translated data with human questions: the manually translated Tamil test questions were machine translated back into English and tested with the original English training data (target language input translation); the manually translated Tamil test questions were also tested with the automatically translated Tamil training questions (creating a target language system).

2.6 Evaluation

We use accuracy as our success measure: the top ranked response to a test question is considered correct if it is identified as a correct response in the linked test data (there are up to 4 correct responses per question). This measure does not take into account non-understanding, that is the classifier's determination that the best response is not good enough (Leuski et al., 2006), since this capability was not implemented; however, since all of our test questions are known to have at least one appropriate response, any non-understanding of a question would necessarily count against accuracy anyway.

Language	Tokenizer / Translation	Stem	Accuracy (%) LM	Accuracy (%) CLM
English	Simple	–	89	82
		+	89	79
	NLTK	–	89	79
		+	89	79
Tamil	Google	–	79	68
		+	71	64
	Manual	–	79	61
		+	68	57

Table 1: Response accuracy on 28 test questions

Question Train	Question Test	Response	Accuracy (%) LM	Accuracy (%) CLM
English	English	Tam (G)	89	82
Tam (G)	Tam (G)	English	79	68
English	Eng (G)	English (NLTK)	79	57
		(Simple)	64	46
Tam (G)	Tam (M)	English	54	43
		Tam (G)	54	39

Table 2: Accuracy with question and response in different languages (G = Google, M = manual)

3 Results

The results of the experiments with matched question and response languages are reported in Table 1. The LM approach almost invariably produced better results than the CLM approach; this is the opposite of the findings of Leuski et al. (2006), where CLM fared consistently better. In most cases, the errors produced by the CLM approach were a superset of those of the LM approach; the only exceptions were Tamil with stemming.

Accuracy of response selection on the Tamil data is about 10% lower than that of English, or twice the errors (6 errors rather than 3). The errors of automatically translated Tamil are a superset of the English errors; however, manually translated Tamil did get right some of the errors of English.

Some of the errors are due to the complexity of Tamil morphology. For example, the following test question receives a correct response in English but incorrect responses in Tamil:

(7) How do you envision the future?
எதிர்காலம் எவ்வாறு கற்பனை செய்கிறிர்கள்

The correct responses are linked to the following training questions.

(8) Are you hopeful about the future?
நீங்கள் எதிர்காலத்தின் மீது நம்பிக்கையாக இருக்கிறீர்களா

(9) Do you have hope for the future?
உங்களுக்கு எதிர்காலத்தின் மீது நம்பிக்கை இருக்கிறதா

In English the word *future*, common to training and test questions, helps identify the desired responses. In Tamil, however, the word “future” appears in distinct case forms: unmarked எதிர்காலம் *etirkaalam* in the test question, but genitive எதிர்காலத்தின் *etirkaalattin* in the training questions. It looks as though some morphological analysis of the Tamil text would be useful. However, while English appears almost invariant to the use of stemming, Tamil performs markedly worse with a stemmer. In this particular case, the stemmer does not unify the *-am* and *-attin* forms, and leaves both forms intact (these forms involve both a stem alternation *-am/-att* as well as a case morpheme *-in*). We are still not able to say why the stemmer hurts performance, but it appears that our application could benefit from a different level of morphological analysis than provided by the current stemmer.

Table 2 reports the results of the experiments which use different languages for the questions and responses. The top two rows use the same language for training and test questions, and only the response language varies. Accuracy is the same as that of the question language: this is necessarily the case for the LM approach, which does not use any of the response content; but it turned out to be the case even for the CLM approach. The middle two rows show the effect of machine translation on the target language inputs: questions in Tamil (manually translated from English) are automatically translated into English, and tested with the original English system. The performance penalty turns out to be the same as for the Tamil systems with matched training and test data, when using the NLTK tokenizer; the simple tokenizer incurs a larger performance penalty. Finally, the bottom two rows show the effect of using machine translation to create a target language system: manually translated questions in Tamil are tested with a system trained on automatic translation from English into Tamil. Performance drops sharply, likely due to mismatches between automatically and manually translated Tamil; this probably speaks to the quality of present state machine translation from English to Tamil. The result means that at present, off-the-shelf machine translation into Tamil is not quite sufficient for a translated dialogue system to work well with human user questions.

4 Discussion

The experiments demonstrate that translating data in the form of linked questions and responses from one language to another can result in a classifier that works in the target language, though there is a drop in performance. The reasons for the drop are not clear, but it appears that simple tokenization is not as effective for Tamil as it is for English, and the level of morphological analysis provided by the Tamil stemmer is probably not appropriate for the task. We thus need to continue experimenting with Tamil morphology tools. The further drop in performance when mixing automatically and manually translated Tamil is probably due to translation mismatches.

Several questions remain left for future work. One possibility is to improve the machine translation itself, for example by adapting it to the domain. Another alternative is to use both languages together for classification; the fact that the manual Tamil translation identified some responses missed by the English classifier suggests that there may be benefit to this approach. Another direction for future work is identifying bad responses by using the distance between question and response to plot the tradeoff curve between errors and return rates (Artstein, 2011).

In our experiments the LM approach consistently outperforms the CLM approach, contra Leuski et al. (2006). Our data may not be quite natural: while the English data are well tested, our sampling method may introduce biases that affect the results. But even if we achieved full English-like performance using machine translation, the questions that Tamil speakers want to ask will likely be somewhat different than those of English speakers. A translated dialogue system is therefore only an initial step towards tailoring a system to a new population.

Acknowledgments

The original English data come from the *New Dimensions in Testimony* project, a collaboration between the USC Institute for Creative Technologies, the USC Shoah Foundation, and Conscience Display.

The second author was supported in part by the U.S. Army; statements and opinions expressed do not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred.

References

Priti Aggarwal, Kevin Feeley, Fabrizio Morbini, Ron Artstein, Anton Leuski, David Traum, and Julia Kim. 2011. Interactive characters for cultural training of small military units. In *Intelligent Virtual Agents (IVA 2011)*, pages 426–427. Springer, September.

Ron Artstein, Sudeep Gandhe, Jillian Gerten, Anton Leuski, and David Traum. 2009. Semi-formal evaluation of conversational characters. In Orna Grumberg, Michael Kaminski, Shmuel Katz, and Shuly Wintner, editors, *Languages: From Formal to Natural. Essays Dedicated to Nissim Francez on the Occasion of His 65th Birthday*, volume 5533 of *Lecture Notes in Computer Science*, pages 22–35. Springer, May.

Ron Artstein, Anton Leuski, Heather Maio, Tomer Mor-Barak, Carla Gordon, and David Traum. 2015. How many utterances are needed to support time-offset interaction? In *Proc. FLAIRS-28*, pages 144–149, Hollywood, Florida, May.

Ron Artstein. 2011. Error return plots. In *Proceedings of SIGDIAL*, pages 319–324, Portland, Oregon, June.

Steven Bird, Edward Loper, and Ewan Klein. 2009. *Natural Language Processing with Python*. OReilly Media Inc.

Bassam Jabaian, Laurent Besacier, and Fabrice Lefèvre. 2010. Investigating multiple approaches for SLU portability to a new language. In *Proceedings of Interspeech*, pages 2502–2505, Chiba, Japan, September.

Bassam Jabaian, Laurent Besacier, and Fabrice Lefèvre. 2013. Comparison and combination of lightly supervised approaches for language portability of a spoken language understanding system. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(3):636–648, March.

Srinivasan Janarthanam, Udhaykumar Nallasamy, Loganathan Ramasamy, and C. Santhosh Kumar. 2007. Robust dependency parser for natural language dialog systems in Tamil. In *Proceedings of 5th IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 1–6, Hyderabad, India, January.

Thomas Lehmann. 1989. *A Grammar of Modern Tamil*. Pondicherry Institute of Linguistics and Culture, Pondicherry, India.

Anton Leuski and David Traum. 2011. NPCEditor: creating virtual human dialogue using information retrieval techniques. *AI Magazine*, 32(2):42–56.

Anton Leuski, Ronakkumar Patel, David Traum, and Brandon Kennedy. 2006. Building effective question answering characters. In *Proceedings of SIGDIAL*, Sydney, Australia, July.

M. Paul Lewis, Gary F. Simons, and Charles D. Fennig. 2015. *Ethnologue: Languages of the World*. SIL International, Dallas, Texas, eighteenth edition. Online version: http://www.ethnologue.com.

S. Lakshmana Pandian and T. V. Geetha. 2008. Morpheme based language model for Tamil part-of-speech tagging. *Polibits*, 38:19–25.

S. Lakshmana Pandian and T. V. Geetha. 2009. Semantic role based Tamil sentence generator. In *International Conference on Asian Language Processing*, pages 80–85, Singapore, December.

Damodharan Rajalingam. 2013. A rule based iterative affix stripping stemming algorithm for Tamil. In *Twelfth International Tamil Internet Conference*, pages 28–34, Kuala Lumpur, Malaysia, August. International Forum for Information Technology in Tamil.

Loganathan Ramasamy and Zdeněk Žabokrtský. 2011. Tamil dependency parsing: Results using rule based and corpus based approaches. In Alexander F. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing: 12th International Conference, CICLing 2011, Tokyo, Japan, February 20–26, 2011, Proceedings, Part I*, volume 6608 of *Lecture Notes in Computer Science*, pages 82–95. Springer, February.

K. Saravanan, Ranjani Parthasarathi, and T. V. Geetha. 2003. Syntactic parser for Tamil. In *Sixth International Tamil Internet Conference*, pages 28–37, Chennai, India, August. International Forum for Information Technology in Tamil.

Christophe Servan, Nathalie Camelin, Christian Raymond, Frédéric Béchet, and Renato De Mori. 2010. On the use of machine translation for spoken language portability. In *Proceedings of ICASSP*, pages 5330–5333, Dallas, Texas, March.

William Swartout, David Traum, Ron Artstein, Dan Noren, Paul Debevec, Kerry Bronnenkant, Josh Williams, Anton Leuski, Shrikanth Narayanan, Diane Piepol, Chad Lane, Jacquelyn Morie, Priti Aggarwal, Matt Liewer, Jen-Yuan Chiang, Jillian Gerten, Selina Chu, and Kyle White. 2010. Ada and Grace: Toward realistic and engaging virtual museum guides. In *Intelligent Virtual Agents (IVA 2010)*, pages 286–300. Springer, September.

M. Thangarasu and R. Manavalan. 2013. Stemmers for tamil language: Performance analysis. *International Journal of Computer Science and Engineering Technology*, 4(7):902–908, July.

David Traum, Kallirroi Georgila, Ron Artstein, and Anton Leuski. 2015. Evaluating spoken dialogue processing for time-offset interaction. In *Proceedings of SIGDIAL*, pages 199–208, Prague, September.

Akhtarul Wasey. 2014. 50th report of the Commissioner for Linguistic Minorities in India. CLM Report 50/2014, Indian Ministry of Minority Affairs.