(ALMOST) ZERO-SHOT CROSS-LINGUAL SPOKEN LANGUAGE UNDERSTANDING

Shyam Upadhyay∗1 Manaal Faruqui2 Gokhan Tür2 Dilek Hakkani-Tür2 Larry Heck†3

1 University of Pennsylvania, Philadelphia, PA
2 Google Research, Mountain View, CA
3 Samsung Research, Mountain View, CA

shyamupa@seas.upenn.edu, {mfaruqui, gokhant, dilekh}@google.com, larry.heck@ieee.org

ABSTRACT

Spoken language understanding (SLU) is a component of goal-oriented dialogue systems that aims to interpret user’s natural language queries in system’s semantic representation format. While current state-of-the-art SLU approaches achieve high performance for English domains, the same is not true for other languages. Approaches in the literature for extending SLU models and grammars to new languages rely primarily on machine translation. This poses a challenge in scaling to new languages, as machine translation systems may not be reliable for several (especially low resource) languages. In this work, we examine different approaches to train a SLU component with little supervision for two new languages – Hindi and Turkish, and show that with only a few hundred labeled examples we can surpass the approaches proposed in the literature. Our experiments show that training a model bilingually (i.e., jointly with English), enables faster learning, in that the model requires fewer labeled instances in the target language to generalize. Qualitative analysis shows that rare slot types benefit the most from the bilingual training.

Index Terms— Spoken Language Understanding, Cross-Lingual, Slot-Filling, Intent Classification

1. INTRODUCTION

Goal-oriented dialogue systems rely on a Spoken Language Understanding (SLU) component to extract meaning from natural language used in conversation [1]. SLU models the semantics of a particular domain by parsing user utterances into semantic frames, which consists of intent and slots. Formally, given the input dialogue utterance \( \vec{x} \) with \( n \) tokens \( \vec{x} = (x_1,x_2,\cdots,x_n) \), the slot filling task involves generating a sequence of \( n \) tags \( \vec{y} = (y_1,y_2,\cdots,y_n) \) which identify the kind and span of different slots, and the intent classification task assigns an intent label \( I \) to the utterance. For example, Fig. 1a shows an utterance and its slots in Begin-Inside-Outside (BIO) encoding with its intent label. To develop spoken dialogue systems in new languages, extending SLU systems to new languages is crucial. The cross-lingual SLU task poses the following problem – given an utterance in another language, the SLU model should generate predictions for slot-filling and intent classification. An example of a Hindi utterance with its slots and intent label is shown in Fig. 1b.

Developing a SLU system for a new language can be quite challenging. While datasets with labeled utterances for training an English model are plentiful, this is not the case for most other languages. Getting high quality human translations is costly for a new

∗Work done while the first author was an intern at Google Research.
†Work done while at Google Research.
training data available in English is translated using the MT system into the target language such that the annotations are preserved after translation. This translated data in the target language now serves as the training data for a new SLU model.

Variants of these approaches have also been proposed, notably the variant of **Test On Source** presented in [4]. [4] made the English SLU model robust to translation inconsistencies by training on MT-distorted English back-translations via the target language. The SLU model was then trained on combination of the original English training data and the back-translated version. The intuition is that the back-translated version will have similar inconsistencies that the translated target language test data will exhibit, allowing the model to adapt to translation errors. We call this approach **Adaptive Test On Source** in our experiments.

A major weakness of both these approaches is that they rely on machine translation. While machine translation is reliable for popular languages like Spanish, German etc., this is not the case for most languages. Indeed, previous work has focused more on high resource languages like Chinese and French, for which high quality machine translation is available. Machine translation also introduces test-time latency in approaches like **Test On Source**. Directly tagging English utterances takes less than 10 ms per query (for our model), while translating from another language to English alone can introduce an order of magnitude larger latency (≈ 100 ms), possibly resulting in reduced conversational experience quality in real use cases. Domain differences also has adverse effects on these approaches, as machine translation models are trained on written parallel text, instead of parallel dialog utterances.

### 3. OUR APPROACH

We first describe a naive model for joint slot filling and intent classification in the target language, which we build on later. This model is inspired by joint slot filling and intent classification approaches from [6, 7] and the success of RNNs on the SLU task [8].

In the following, we denote a training example in English as \(x^e, y^e, L^e\) and an example in target language using \(x^f, y^f, L^f\). Let \(\{(x^e, y^e, L^e)\}^M\) denote English training data with \(M\) examples, and \(\{(x^f, y^f, L^f)\}^N\) denote target language training data with \(N\) examples, where \(M > N\). We use \(\Phi(x_i)\) to denote embedding of a token \(x_i\) and \(\Phi(\bar{x})\) as shorthand for \(\Phi(x_1), \Phi(x_2), \ldots\).

**Naive Model.** The naive model uses \(\{(x^f, y^f, L^f)\}^N\) to train a bidirectional RNN to predict both the intent and BIO tags (shown in Fig. 3). The hidden state at each timestep is used to predict the corresponding BIO tag, and the last hidden state is used to predict the intent label. Formally,

\[
\vec{h} = \text{BiRNN}(\Phi(\bar{x})) \\
\frac{y_i}{\text{Softmax}(W_\phi \vec{h}_i)}, \quad L = \text{Softmax}(W_L \tilde{h}_n)
\]

where \(\vec{h}\) is the sequence of hidden states generated by the concatenation of forward and backward outputs from the bidirectional RNN \(\text{BiRNN}\). \(\tilde{h}_n\) is the last hidden state, \(W_\phi\) and \(W_L\) are model weights, and \(\text{Softmax}\) is the softmax operation. The learning objective is the sum of the sequence-tagging loss and the intent classification loss, \(L_{\text{naive}} = L_{\text{seq}}(\tilde{y}, y^f) + L_{\text{clf}}(L, y^f)\), where \(\tilde{y}\) and \(L\) are current model predictions, averaged over all training examples. The model parameters, including the word embeddings \(\Phi(x_i)\), are learnt during training.

The naive approach only utilizes the little training data that might be available in the target language. However, for most SLU tasks, training data in English is available, therefore it is desirable to use it for improving generalization in a new language. We show how to achieve this by training a joint model for both languages, such that parameters are shared across languages.

**Bilingual Embeddings.** To encourage parameter sharing we need to ensure the features (viz. the word embeddings) in different languages lie in the same vector space. However, word embeddings trained monolingually for two different languages do not encode cross-lingual semantics appropriately. For instance, the embeddings for \(\Phi_e(\text{atlanta})\) and its Hindi translation \(\Phi_f(अटलांटा)\) need not have high cosine similarity. To achieve this, we first align the embeddings into a shared vector space.

Aligning word embeddings in different languages has been a popular research direction in natural language processing [9, 10, 11, 12, 13, inter alia]. A common approach is to learn linear transformations \(W\) and \(V\), such that vectors for semantically equivalent words are aligned (for instance, \(W \Phi_e(\text{atlanta})\) and \(V \Phi_f(अटलांटा)\) will have higher cosine score) and reside in a shared vector space, which we denote as \(\Phi_{e,f}\). We adopt this simple approach and use publicly available embeddings from [14] with the alignment matrices from [15] to project embeddings into a shared space. We also experimented with other approaches for aligning embeddings like CCA [16], but got the best results using off-the-shelf vectors.

#### 3.1. Zero-Shot SLU

Aligned word embeddings also enable zero-shot SLU. For this, we first train an English SLU model on \(\{(x^e, y^e, L^e)\}^M\) using \(\Phi_{e,f}\) to embed English tokens. To ensure the embeddings remain aligned across languages, they are not updated during training.

The model is then directly tested on the target language test utterances, using \(\Phi_{e,f}\) to embed the target language tokens. As \(\Phi_{e,f}\) ensures embeddings from different languages for semantically equivalent words are similar, the model parameters can still predict certain slots accurately. The approach is shown in Fig. 3, where the parameters enclosed in the grey box are pre-trained on English.
split of 4978 utterances as supervision in English. Automatic translations for the TEST ON SOURCE and TRAIN ON TARGET approach were generated using Google Translate.

**Evaluation and Training Setup.** We compare the naive and the bilingual models by varying the amount of training data available in the other languages. We also include the TRAIN ON TARGET and TRAIN ON SOURCE approaches in the slot filling comparison to demonstrate their shortcomings.

We plot the evaluation metric (tagging F1 or intent accuracy) against the number of training examples. When using a fraction of the training data with the naive and the bilingual models, we sub-sample 5 times from the entire pool of training examples, and report the average performance. We use the standard conlleval script [21] for evaluating the slot-filling, and classification accuracy for intent.

In all experiments, we used size 300 embeddings from [14, 15], normalized to unit norm. The RNN unit was a LSTM [22], with hidden state of size 100. The language indicator vector \( k \) of size 5 in the bilingual model was trained along with model parameters. All models were trained for a total of 10 epochs with a batch size of 5, using Adam [23] with a learning rate of 1e-3, and a word dropout rate of 0.5 [24]. All models were implemented using Tensorflow [25].

### 4.1. Experimental Results

The slot filling results are shown in Figure 5. TRAIN ON TARGET is worse on Hindi due to poor translation quality, even though it had \( \approx 5k \) queries to train on. TEST ON SOURCE performs quite well on both languages, and improves substantially after adding adaptive training with back-translations, as shown by the ADAPTIVE TEST ON SOURCE curve. This is reasonable as translation quality is higher when translating from a foreign language to English (viz. TEST ON SOURCE) than the opposite direction (viz. TRAIN ON TARGET). In comparison, the zero-shot approach with a trained English SLU performs quite well, demonstrating their shortcomings.

For the naive model, we approximately need around 500 examples in both languages to achieve a F1 of 75.0. In fact, it beats the previous best approach, ADAPTIVE TEST ON SOURCE, with only 100 examples in both languages. Note that
We proposed a simple bilingual training approach to train a SLU model in a new language jointly with English, without relying on machine translation. Our approach outperforms existing state-of-the-art approaches on new SLU benchmarks\textsuperscript{2} in Hindi and Turkish, while maintaining competitive performance on English.

There are several avenues of future research. More parameter sharing can be achieved across languages by using character level embeddings in conjunction with word embeddings. A fully multilingual approach which trains the same model to handle three or more languages is also a natural extension of our work.

\textsuperscript{1} training only on English achieves 95.2.

\textsuperscript{2}Available at github.com/google-research-datasets/dialogue/tree/master/multilingual-atis
6. REFERENCES


