Language-Agnostic and Language-Aware Multilingual Natural Language Understanding for Large-Scale Intelligent Voice Assistant Application

Daniel (Yue) Zhang, Jonathan Hueser, Yao Li, Sarah Campbell
Alexa AI, Amazon
Cambridge, MA, USA
{dyz, hueserjh, mznyao, srh}@amazon.com

***Abstract*—Natural language understanding (NLU) is one of the most critical components in goal-oriented dialog systems and enables innovative Big Data applications such as intelligent voice assistants (IVA) and chatbots. While recent advances in deep learning-based NLU models have achieved significant improvements in terms of accuracy, most existing works are monolingual or bilingual. In this work, we propose and experiment with techniques to develop *multilingual* NLU models. In particular, we first propose a purely language-agnostic multilingual NLU framework using a multilingual BERT (mBERT) encoder, a joint decoder design for intent classification and slot filling tasks, and a novel co-appearance regularization technique. Then three distinct language-aware multilingual NLU approaches are proposed including using language code as explicit input; using language-specific parameters during decoding; and using implicit language identification as an auxiliary task. We show results for a large-scale, commercial IVA system trained on a various set of intents with huge vocabulary sizes, as well as on a public multilingual NLU dataset. We performed experiments in explicit consideration of code-mixing and language dissimilarities which are practical concerns in large-scale real-world IVA systems. We have found that language-aware designs can improve NLU performance when language dissimilarity and code-mixing exist. The empirical results together with our proposed architectures provide important insights towards designing multilingual NLU systems.**

***Index Terms*—natural language understanding, multilingual representation, intelligent voice assistant**

I. Introduction

Recent years have seen the increasing popularity of feature-rich intelligent voice assistants (IVAs) such as Amazon Alexa, Google Assistant, Apple Siri, and more. In 2020, IVAs were used in over 4 billion devices in the world, and are estimated to reach 8.4 billion devices by 2024 [1]. A key enabling technology for IVA is natural language understanding (NLU) which is the task of extracting the intents and semantics from user queries [1], [2]. NLU in IVAs typically consists of the following sub-tasks: intent classification (IC) and slot filling (SF). Consider an example query from an IVA customer: "What is the airfare between Denver and Pittsburgh?". The IC sub-task identifies "*airfare*" as the speaker's intent. Meanwhile, the SF sub-task labels "Denver" and "Pittsburg" as "*B-fromloc.city_name*" and "*B-toloc.city_name*" respectively, which provide important context information (i.e., departure and destination cities) associated with the intent. The IC and SF tasks' outputs jointly provide structured information to guide the proper response towards the user's query, such as replying with the available flights and airfares between the two cities.

The majority of previous NLU research has focused on improving the performance of NLU for a specific language (i.e., monolingual), and has achieved significant advances in improving accuracy and efficiency [3], [4], [5], [6]. However, the monolingual approaches require deploying one NLU model per language and therefore cannot easily scale to support billions of IVA customers who speak different languages across the globe. Inspired by the success of multilingual language representations such as mBERT [7] and domain adaptation techniques [8], recent works have started investigating potential ways to expand cross-lingual transferability for NLU [9], [10], that allow deploying NLU for a low resource language with few or even no training data. However, these methods focus only on bilingual cases and only explored language-agnostic model designs. In this paper, we introduce multilingual capabilities for NLU by leveraging finetuning pretrained multilingual representation and propose novel language-agnostic and language-aware designs for multilingual NLU.

While building multilingual models has been widely studied in other NLP tasks such as machine translation (MT) and question answering (QA), achieving multilingual NLU has some unique challenges. First, existing research in cross-lingual NLU uses purely language-agnostic designs, where a shared off-the-shelf pretrained encoder, as well as a shared decoder, are applied for all languages [9]. However, such purely language-agnostic design is questionable as recent evidence has shown that a rigid share-all layout provides little flexibility in handling language diversity [11], and leads to degraded performance when applied to dissimilar languages that lack common lexical features [12]. A potential solution to address language differences is to add language-specific capacity to the multilingual models by introducing separated model parameters, such as language-specific encoders [11] or language-specific output layers [13]. However, such designs can lead to significantly larger model complexity and can be impractical for IVA systems

[1]https://www.statista.com/statistics/973815/worldwide-digital-voice-assistant-in-use/

978-1-6654-3902-2/21/$31.00 ©2021 IEEE

that are commonly deployed on resource-constrained devices, and have strict latency requirements [14], [15].

Second, code-mixing and code-switching are very common in multilingual IVA systems where the customers' queries can be composed of multiple languages [16], which pose extra complexity to incorporate language awareness into multilingual NLU models. Existing language-aware multilingual modeling research often assumes each input is associated with a language code and may fail to handle queries where the language classification is ambiguous. Consider an example English-Japanese bilingual model where each language is processed by its own set of parameters (e.g., [11]). Then, for a query that is mixed with both Japanese and English, it will be problematic if the utterance is processed by either the Japanese- or English-specific parameters.

In short, the design choice for multilingual NLU, especially in terms of language-agnostic vs. language-aware multilingual modeling approaches, is not clear and has not been studied in prior NLU research. It is also not clear how different multilingual NLU model designs can handle practical concerns in global-scale IVA systems including language dissimilarity and code-mixing.

In this work, we first propose a language-agnostic multilingual NLU model design that uses pretrained multilingual BERT to extract multilingual sentence-level and token-level representations, followed by domain adapted joint IC-SF decoding tasks. A novel co-appearance regulation technique is further proposed to enforce consistency in the IC and SF outputs. Then, three alternative language-aware multilingual NLU designs are proposed that explore different ways to address language differences and code-mixing in real-world IVA traffic. The proposed designs are evaluated on two real-world datasets: a sampled large-scale commercial IVA system's live traffic and a publicly available NLU dataset. The empirical results have shown that the language-aware multilingual NLU designs have achieved the best performances in languages with linguistic dissimilarities while the proposed language-agnostic design is superior in linguistically similar languages. Moreover, we found that the language-aware design with explicit token-level language labels has achieved on par or better performance as compared to the language-agnostic counterpart in code-mixed queries.

To the best of our knowledge, this is the first work that explores both language-agnostic and language-aware designs for multilingual NLU and performs systematic comparisons regarding their trade-offs. These findings provide important insights into the design choices towards multilingual NLU.

II. Related Work

A. Natural Language Understanding

Intent classification and slot filling are two fundamental tasks in natural language understanding [17], [18]. To date, deep neural network based approaches, including LSTM and Transformers, have been the *de facto* model choices for state-of-the-art NLU systems [3], [4], [5], [6]. While most of the recent advances in NLU systems are monolingual models, some prior works have been developed to enable the cross-lingual capability for NLU systems. For example, Thi Do and Gaspers have proposed a simple but effective weight transfer approach that first trains on the source language (English) with subsamples from the target language (Germany) as a tuning set. Then the trained weights are utilized to initialize training parameters for target language data [10]. Xu *et al.* further proposed two transfer learning approaches by i) translating source language to a low-resource target language; and by ii) finetuning a pretrained mBERT model [9]. However, these works focus on bilingual cases and zero-shot transfer learning for low-resource target languages. In contrast, our research focuses on the more generic case of developing a joint model for multilingual languages. The work from Castellucci *et al.* [19] is probably most relevant to our paper, where the authors proposed a multilingual NLU model with joint IC-SF decoding using pretrained mBERT model. However, this work only explores language-agnostic designs with simple off-the-shelf pretrained models. The proposed work, on the other hand, explores both language-agnostic and language-aware designs and systematically studies their trade-offs.

B. Multilingual Modeling and Linguistic Dissimilarity

Multilingual language representations using large-scale pretrained encoders such as BERT and GPT3 learned from massive text corpora in general domains have brought significant performance gains in various multilingual NLP tasks [7]. However, the representations learned from these pretrained encoders are universal (i.e., "language-agnostic") and the direct application of such encoders raises many concerns in multilingual systems. For example, Yang *et al.* [12] and Lin *et al.* [20] found that for closely related languages with similar alphabets, it is beneficial to train a multilingual model with a shared encoder and decoder for a sequence labeling task, but less effective when applied to dissimilar languages that lack common lexical features. Several recent works have introduced mechanisms to force language-specific behavior in multilingual models. For example, Zhang *et al.* introduced a gating layer after every transformer sub-layer in the encoder and the decoder that learns to route every input through either a language-specific projection layer or a language-agnostic projection layer [11] for Neural Machine Translation (NMT) tasks. Sen *et al.* proposed an alternative approach where a shared language-agnostic encoder is used followed by language-specific decoders for NMT [13]. These prior works indicate that the design choices for language-specific vs. language-agnostic designs are quite task-specific and not intuitive especially for the multilingual NLU case where no systematic comparisons have been done in prior research. In this paper, we propose various language-specific and language-agnostic model structures to explore optimal settings for multilingual NLU.

C. Multilingual Modeling and Code-Mixing

The issue of code-mixing (CM) in multilingual modeling is relevant to our work as well. Various efforts have been proposed to improve domain adaptation of multilingual models

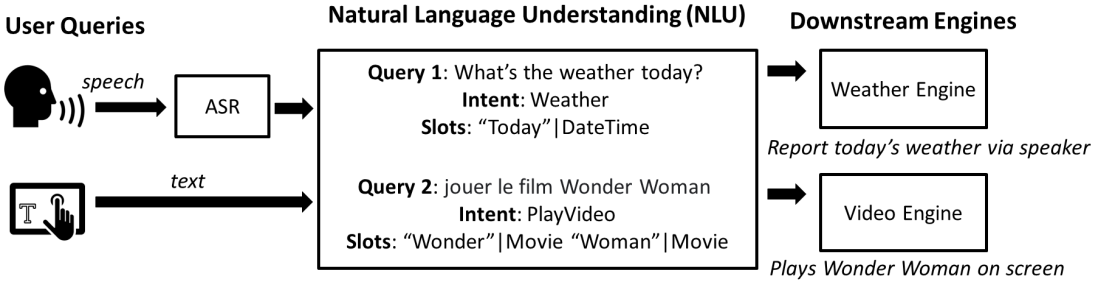

Fig. 1: Illustration of NLU in IVA System

to handle CM data. Most of these works focus on performing data augmentation to generate CM data for model training. For example, Samanta *et al.* developed a variational autoencoder-based generative model to synthesize CM texts [21]. Gao *et al.* proposed an alternative CM generation approach using BERT and Generative Adversarial Networks (GAN) [16]. Both methods have shown significantly improved perplexity in language models. It is also observed that off-the-shelf mBERT-based multilingual representation can generalize and transfer across scripts and to code-switching fairly well [22]. Additionally, Santy *et al.* performed a systematic analysis of mBERT model performance on various NLP tasks with CM data. They found out that using naturally occurring code-mixed data brings the best performance improvement after finetuning [23]. However, it has not been studied whether a language-agnostic mBERT-based representation can handle CM scenario well in the multilingual NLU task. In this paper, we propose an alternative language-aware multilingual NLU design that explicitly considers CM, and compares it with its language-agnostic counterpart in handling CM.

III. Problem Definition

In this section, we first provide a brief overview of the key terms and concepts in NLU and then formally define the multilingual NLU problem we address in this paper.

A. Natural Language Understanding and IC-NER Tasks

DEFINITION 1. Natural Language Understanding (NLU): The task of NLU is to produce structured insights from free-form task-oriented dialogs. NLU in IVA systems is sometimes used interchangeably with Spoken Language Understanding (SLU) [24] to highlight the nature of speech input. In a typical IVA system illustrated in Figure 1, users provide input queries that are either speech signals via microphone or texts via keyboard or touchpads. The speech signals are transcribed into texts by a subsequent Automatic Speech Recognition (ASR) module. NLU then takes both the input texts and transcribed texts and performs two core tasks - IC and SF tasks, that generate user intents and slots respectively. The generated intents and slots are then fed into downstream engines to generate proper responses to the user such as playing music/movie or answer a user's question.

DEFINITION 2. User Query (Utterance): the user query is defined as an IVA user's question (e.g., "what is the weather like today") or instruction (e.g., "set a reminder in 5 minutes"). It is also referred to as an "utterance" in NLU research. We use the term "user query" and "utterance" interchangeably in this paper. Formally, we define the input utterance of length T as:

$$u = (t_1, t_2, ..., t_T) \tag{1}$$

where $t_i, i \in [1, T]$ denotes the i^{th} token of the utterance. We further use U to denote a set of customer utterances to the IVA system.

DEFINITION 3. Intent Classification (IC): The task of intent classification is the automated categorization of text data based on customer goals. Formally, the goal of IC is to find:

$$\arg\max_{\tilde{I}} Pr(I = \tilde{I}|u), \forall u \in U \tag{2}$$

where $\tilde{I}$ and I are the predicted intent label and actual user intent, respectively.

DEFINITION 4. Slot Filling (SF): The task of slot filling is to extract the values of certain types of attributes relevant to the customer's intent to provide enough context for the IVA to respond. For example, given a customer intent of "play music", it is SF's job to extract the name of the music to be played. Formally, the goal is to find:

$$\arg\max_{\tilde{S}} Pr(S = \tilde{S}|u), \forall u \in U \tag{3}$$

where $\tilde{S} = (\tilde{s}_1, \tilde{s}_2, ..., \tilde{s}_T)$ are the predicted slot labels for each token in u, and $S = (s_1, s_2, ..., s_T)$ are the ground truth slot labels for u.

B. Multilingual NLU

This paper extends the traditionally monolingual IC-SF tasks of NLU to a multilingual setting, that assumes a joint multilingual NLU model will be able to process user queries from various countries and in multiple different languages. The goal of this paper is to design and compare both language-agnostic and language-aware approaches defined below that optimize the IC and SF tasks' performances in multilingual NLU.

DEFINITION 5. Language-Agnostic NLU: In language-agnostic multilingual NLU, the model does not distinguish the input utterances in terms of their languages. The model also does not have any language-specific parameters.

DEFINITION 6. Language-Aware NLU: In language-aware multilingual NLU, an utterance in different languages will be treated differently in the NLU model. In general, the model either 1) explicitly uses the language code of the input utterances, 2) has language-specific parameters; or 3) is trained to be able to distinguish languages of input utterances. It is also possible that a language-aware NLU model uses a combination of all these approaches.

The proposed language-agnostic and language-aware multilingual NLU designs are presented in Sections IV and V.

IV. Proposed Language-Agnostic Multilingual NLU Frameworks

In this section, we present a language-agnostic approach to multilingual NLU.

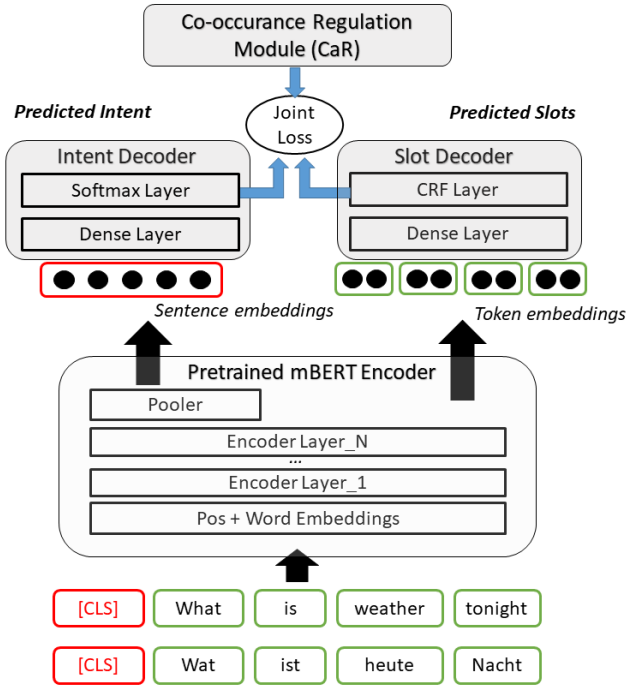

Fig. 2: Language-Agnostic Multilingual NLU Architecture

A. Language-Agnostic Multilingual NLU (Agno)

The language-agnostic multilingual NLU design (*Agno* for short) is illustrated in Figure 2. The model has three major components - 1) a pretrained multilingual BERT (mBERT) encoder, 2) a joint task IC-SF decoder, and 3) a novel co-appearance regularization (CaR) module. The model architecture is illustrated in Figure 2.

Pretrained Encoder. We use a distilled mBERT model as the pretrained encoder to extract multilingual representations from input utterances. The encoder is distilled from mBERT-Large [2] and has 768-dimensional word and position embedding vectors that are fed to a 12-layer, 4-head BERT encoder with a 768 hidden size and a 0.1 dropout rate. The distillation was performed based on the method described in [25].

Given an input utterance of length T: $u = (t_1, t_2, ..., t_T)$, we add a special classification token t_0 or $[CLS]$ token at the beginning of the input following Devlin *et al.* [7]. Next, the pretrained encoder is used to produce a sequence of contextualized representations $h = (h_0^i ... h_T^i)$ for the input sequence; where h_t^i represents the hidden vector for the t^{th} input token from the i^{th} encoder layer, $i \in [1, K]$. We use K=12 as total number of layers in our pretrained mBERT encoder. We take two outputs from the generated hidden vectors: 1) a sentence embedding E_s output from the BERT pooler by applying linear transformation and tanh activation on the $[CLS]$ embeddings of each layer, namely $E_s = \text{BertPool}(h_0^1, h_0^2, ..., h_0^K)$; and 2) token level embeddings E_w for each word in the input utterance, namely $E_w = (h_1^K, h_2^K, ..., h_T^K)$.

For ease of notation, we omit K and use $E_s = h_0$, and $E_w = (h_1, h_2, ..., h_T)$ to denote the sentence embedding and token embedding of input utterance u.

Joint IC-SF Decoder. For intent classification, we implement an IC decoder that takes E_s from the pretrained encoder as the sequence representation, and then apply a dense layer and a softmax function to predict the intent probability. Let Θ^I denote the parameters of the IC decoder, and Θ denote the mBERT encoder's parameters, formally we have:

$$\begin{aligned} \textbf{Dense Layer:} \quad & \tau^I = W^I \cdot h_0 + b^I \\ \textbf{Softmax Layer:} \quad & p(\tilde{I}|u) = \text{softmax}(\tau^I) \propto \exp(\tau^I) \\ \textbf{IC Loss:} \quad & \mathcal{L}(\Theta, \Theta^I) = -\sum_{u \in U} log\ p(\tilde{I}|u) \end{aligned} \tag{4}$$

where τ^I and (W^I, b^I) denote the output and parameters of the dense layer, respectively.

For slot filling task, we implement a SF decoder that takes the token embeddings $(h_1, h_2, ..., h_T)$ from the mBERT encoder, and then apply a dense layer followed by a Conditional Random Field (CRF) decoder. Let Θ^s denote the parameters of the SF decoder, formally we have:

$$\begin{aligned} \textbf{Dense Layer:} \quad & \tau^S = W^S \cdot (h1, h2, ..., h_T) + b^S \\ \textbf{CRF Layer:} \quad & p(\tilde{S}|u) = \text{CRF}(\tau^S) \\ & \propto \exp\left(\sum_{i=1}^{T} \mathcal{T}_{\tilde{s}_i, \tilde{s}_{i-1}} \times \tau^{\tilde{s}_i}, \tilde{s} \in \tilde{S} \right) \\ \textbf{SF Loss:} \quad & \mathcal{L}(\Theta, \Theta^S) = -\sum_{u \in U} log\ p(\tilde{S}|u) \end{aligned} \tag{5}$$

where τ^S and (W^S, b^S) denote the output and parameters of the dense layer, respectively. $\mathcal{T} \in \mathbb{R}^{|S| \times |S|}$ is the transition matrix recording the transition probability between two SF labels, where $|S|$ is the total number of distinct SF labels.

Following [10] and [6], the IC and SF decoders are trained together as a multi-task model by minimizing a joint IC-SF loss function: $\mathcal{L}(\Theta, \Theta^I, \Theta^S) = \mathcal{L}(\Theta, \Theta^I) + \mathcal{L}(\Theta, \Theta^S)$. The joint loss is further penalized by a novel co-appearance regularization introduced next.

Co-appearance Regularization (CaR). We introduce a novel co-appearance regularization (CaR) module to alleviate the contradicting IC-SF output issue observed in joint IC-SF training. Consider an example utterance "What's the weather in London" where the IC output is "weather" (correct), and SF output is "what's|O the|O weather|O in|O London|from_city" (incorrect). "O" stands for default SF label - "Other". In this example, the IC output and SF tokens are contradicting because the SF token "from_city" should never appear in the "weather" intent and is exclusive to the "airfare" intent.

In a sense, there should exist constraints that certain SF labels can only be associated with certain intents. We refer such constraints as *co-appearance constraints*. Such constraints cannot be satisfied in existing joint IC-SF training approaches [10], [6] due to lack of dependence: IC is unaware of the predicted SF label and vice versa. To incorporate the co-appearance constraints into the model, the CaR module deploys a special sequence tagging task that allows the SF decoding process to be explicitly aware of intent-exclusive SF labels. In this task, we modify the SF training data as following: for each token, instead of labeling its corresponding SF tag, we directly label the intent that it exclusively belongs to. For tokens with slots that belong to multiple intents, we neglect such tokens (illustrated in Figure 3). CaR module is trained to correctly predict these intent-exclusive tokens of each utterance. The regulation is achieved by sharing the token embeddings from the mBERT encoder $(h1, h2, ..., h_T)$ together with SF decoding. The intuition of this design is to force the model to "pay attention" to these intent-exclusive tokens in each utterance, as they provide a direct hint of which intent it belongs to, and penalize the SF loss if it fails to. We illustrate the input and model details of CaR in Figure 3.

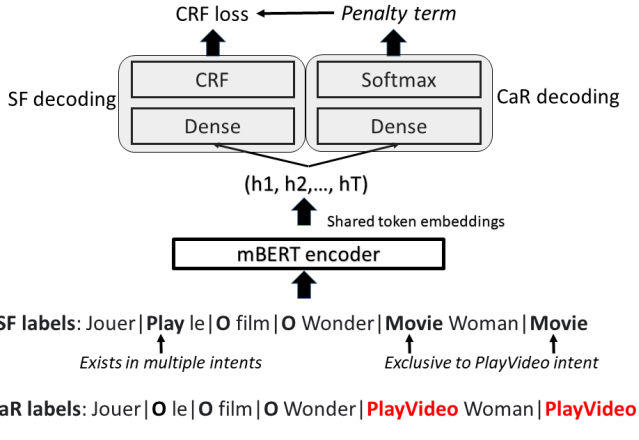

Fig. 3: Illustration of CaR-regulated SF Training

Note that while the CaR decoding process is similar to SF decoding, a softmax layer is used instead of a CRF layer. This is due to the fact that the transition probability in CRF is not meaningful in the CaR. The softmax loss is used as a penalty term to regulate the CRF loss to enforce concurrence constraints. Let $\mathcal{L}(\Theta, \Theta^C)$ denote the penalty term of the CaR module, the loss of the entire model can be formally derived as: $\mathcal{L}(\Theta, \Theta^I, \Theta^S, \Theta^C) = \mathcal{L}(\Theta, \Theta^I) + \mathcal{L}(\Theta, \Theta^S) + \lambda^C \mathcal{L}(\Theta, \Theta^C)$, where λ^C is a tunable weight for the CaR penalty term.

Training. The training process of the proposed *Agno* model involves finetuning the entire decoder blocks and part of the mBERT encoder via a gradual unfreezing technique. The finetuning process is discussed in detail in Section VI.

Inference. During the inference process the IC and SF decoders' outputs are first collected to get initial intent and slot labels. Then the CaR output is collected with the same input utterance. Note that the CaR output also contains predicted intent information. The outputs of the three components are then ensembled through an arbitration process illustrated in Figure 4. In specific, the IC output will be overridden with the predicted intent from CaR if two conditions are satisfied: 1) IC output disagrees with both CaR and SF, and 2) CaR output is consistent with SF output.

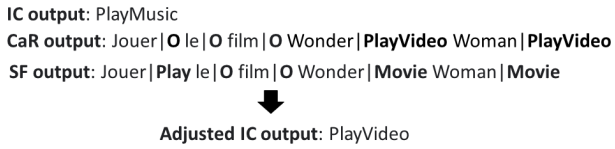

Fig. 4: Illustration of the arbitration process. In this example, the IC output (PlayVideo intent) is contradicting to both CaR and SF outputs (PlayVideo and Movie tags). The IC output is therefore replaced with PlayVideo intent.

In the subsequent sections, we use *Agno* to represent the complete language-agnostic design including the IC-SF joint task and CaR module. As an ablation study, we further add a candidate *Agno_noCaR* to independently evaluate the effectiveness of the CaR module.

V. Proposed Language-Aware Multilingual NLU Frameworks

In this section, we propose three distinct language-aware architecture variants to explicitly enable language-specific parameters in the encoding and/or decoding process of multilingual NLU.

A. Model Design Considerations

The design choices of language-aware multilingual NLU models are based on the following two considerations:

Fairness. In order to be fairly comparable to the language-agnostic approach, the language-aware candidates' model settings are as similar to *Agno* as possible. In particular, we use the exact same pretrained mBERT encoder prior to finetuning and reuse the joint IC-SF decoding process as well as the CaR module for all language-aware candidates. We also intentionally exclude design options where significantly

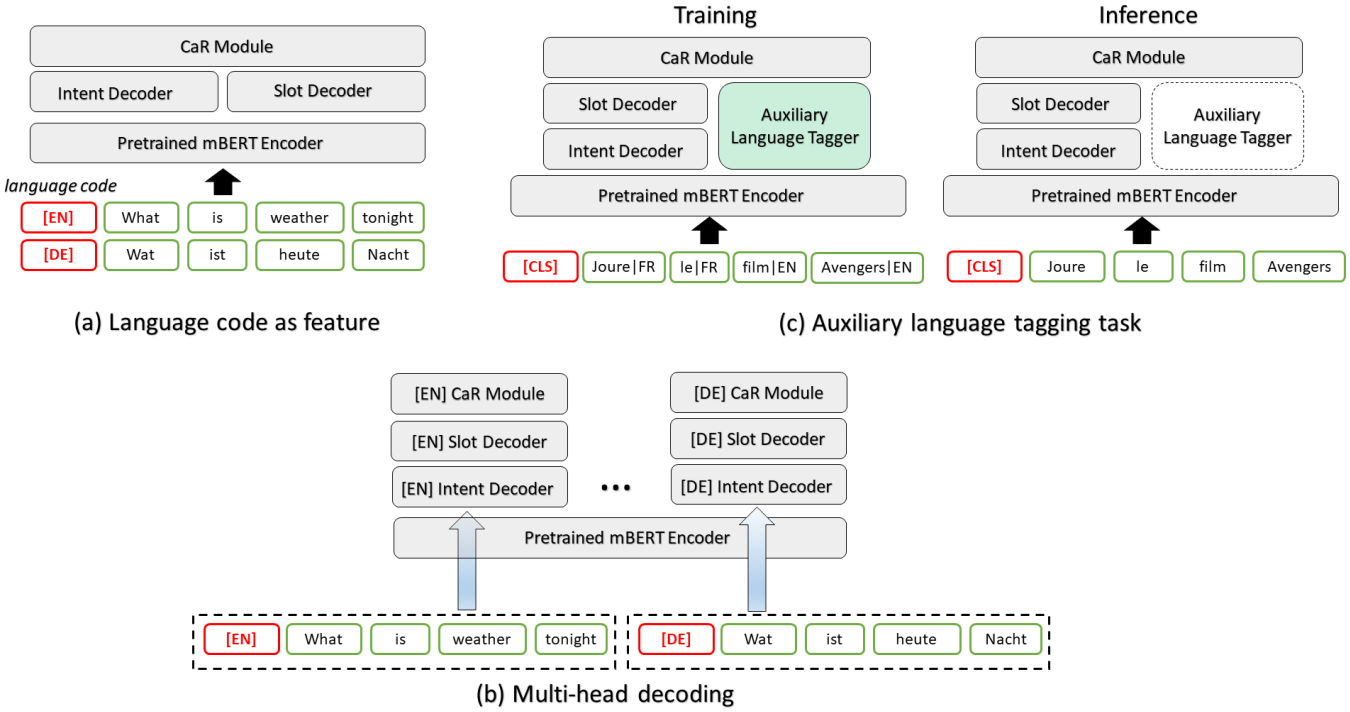

Fig. 5: Language-Aware Multilingual NLU Architectures

larger model capacities (thus unfair) are used for language-aware candidates such as stacking individual encoders for each language [11].

Diversity. Since the design choice of language-aware multilingual NLU is not clear and has not been studied in previous research, it makes sense to explore a diverse set of approaches. In the proposed language-aware multilingual NLU models, we explore three distinct approaches toward achieving language awareness that will be presented in detail in the following subsections.

1) Language-awareness achieved by using explicit language code as input feature ($LA_{feature}$)
2) Language-awareness achieved by independent language-specific parameters per language via a multi-head decoding design (LA_{mh})
3) Language-awareness achieved by implicit language detection as auxiliary task (LA_{aux})

B. Language Code as Input Feature ($LA_{feature}$)

The first design to incorporate language-awareness into multilingual NLU is the most simplistic approach - by adding an additional language code to the input utterance as illustrated in Figure 5-(a). This model is referred to as "$LA\_feature$". In particular, we override the $[CLS]$ token with a language code (e.g., $[EN]$, $[DE]$, $[FR]$, etc.). These language codes are special tokens and need to be added into mBERT's vocabulary. By replacing the $[CLS]$ token, the embedding of the language code from each mBERT encoder layer will be used by the mBERT's pooler and generate sentence embedding. Therefore, the IC decoder will be explicitly using the language code as a feature to generate intent predictions. This approach is simple and straightforward. A potential limitation is that the language code is assumed to be available during inference time, which may not be practical in a real-world setting. This can be mitigated by using language identification (LID) models to automatically generate language codes for input utterances which we will be leveraging in the evaluation process.

C. Language Routing with Multi-head Decoding (LA_{mh})

We further present a multi-head decoding design for language-aware multilingual NLU. We refer to this design as "LA_mh". In this design, the encoder stays language-agnostic, while each language has its own IC-SF decoder and CaR module as illustrated in Figure 5-(b). All decoders share the same output embeddings from the encoder. A language code similar to $LA\_feature$ is added to the input utterance and an incoming utterance will be hard routed to one of the decoders corresponding to their language code during inference. During training, the updated gradients from an utterance will only be used to update its corresponding decoders' loss functions. For example, an English utterance will only be used to update gradients to the English decoders' losses and will not affect the decoders for Germany. The intuition of this approach is to allow dissimilar languages to have their own specific set of parameters during the decoding process.

D. Language Tagging as Auxiliary Task (LA_{aux})

Finally, we propose an auxiliary task design where we introduce an extra token-level language tagging (LT) decoder to

TABLE I: Summary of Proposed Candidates

	Name	Language-Aware Encoder?	Language-Aware Decoder?	Language Specific Parameters?	Co-appearance Constraints?	Language Code Required?	Handle Code Mixing?
Language-Agnostic Candidates	$Agno$	x	x	x	✓	x	x
	$Agno\_noCaR$	x	x	x	x	x	x
Language-Aware Candidates	$LA\_feature$	✓	✓	x	✓	training and inference	x
	LA_aux	✓	✓	x	✓	training only	✓
	LA_mh	x	✓	✓	✓	training and inference	x

TABLE II: Dataset Statistics

	Data Per Language			Languages	Domains
	Train	Dev	Test		
MultiATIS++	4,488	490	893	EN, DE, FR, ES, PT, HI*, JA	Flight
Commercial VA	400k	40k	80k	EN, DE, FR, ES, IT	Music, Video, Books, SmartHome, Weather

*In MultiATIS++, Hindi (HI) has a split of 1,440(train)/160(dev)/893(test) utterances due to data sparsity.

the model (referred to as "LA_aux"). The model is illustrated in Figure 5-(c). The LT decoder classifies the language code of each token for a given input utterance. In particular, the LT decoder takes the token embedding $(h1, h2, ..., h_T)$ from the pretrained mBERT encoder, then applies a feed-forward network followed by a CRF layer. Similar to the joint IC-SF design, the auxiliary LT decoder is finetuned together with other decoders by minimizing a joint loss:

$$\mathcal{L}(\Theta, \Theta^I) + \mathcal{L}(\Theta, \Theta^S) + \lambda^C \mathcal{L}(\Theta, \Theta^C) + \lambda^{LT} \mathcal{L}(\Theta, \Theta^{LT}) \quad (6)$$

where $\mathcal{L}(\Theta, \Theta^{LT})$ is the loss for the LT decoder and λ^{LT} is a tunable weight for the LD decoder loss.

The design intuition of choosing a fine-grained token-level LT decoder is to explicitly address code-mixing (CM) scenarios where an input utterance can contain multiple languages. For example, consider a code-mixed query "John Lennon 音楽を再生するす哉 (play John Lennon's music)". The [EN] SF decoder in LA_mh may not handle the Japanese portion of the query well. Similarly [JA] SF decoder in LA_mh can mis-annotate the English portion of the utterance. In contrast, by minimizing the loss of the LD decoder, LA_aux's shared mBERT encoder is able to distinguish the language of each token in the input utterance while predicting SF labels.

Another advantage of LA_aux is that during inference time, the LT decoder is ignored and no language code is needed. We illustrate the difference between training input and inference input in Figure 5-(c). We observed that most code-mixing happens when English is mixed with another language. To acquire the labeled training data, we use a simple but effective dictionary-based approach [26] to annotate common English words in each non-English utterance.

VI. Experiments

In this section, we conduct comprehensive comparisons of proposed multilingual NLU models to answer to following research questions of interest:

- **Q1:** Which multilingual NLU candidate achieves the best overall performance on the IC-SF task?
- **Q2:** Do language-aware NLU candidates perform better in code-mixing cases?
- **Q3:** What are the training time and model complexity trade-offs for each candidate?

The experiment setup is explained in detail below.

A. Experiment Setup

Datasets. We use two real-world multilingual NLU datasets to perform training and evaluation: 1) **MultiATIS++**: a popular public multilingual NLU dataset. It originally contained 5,871 English utterances from ATIS (Air Travel Information Services) corpora, and was further extended to support multiple languages via professional native translators [9]. The dataset contains subsamples for 7 languages - English (EN), French (FR), German (DE), and Spanish (ES), Portuguese (PT), Hindi (HI), and Japanese (JA). 2) **Commercial VA**: we use a random snapshot of a curated set of utterances taken from the live production traffic of an anonymous commercial VA system. The data is carefully preprocessed so that users are not identifiable. We randomly sampled 520k annotated sentences for each language from 5 domains: Music, Video, Books, Weather, and SmartHome. The dataset contains subsamples for four languages - English (EN), French (FR), Germany (DE), and Spanish (ES). We summarize the two datasets in Table II.

Metrics. We evaluate the performance of multilingual NLU candidates using semantic error rate (SemER) [27], a metric that combines errors from IC and SF tasks. Formally, the SemER metric is defined as

$$SemER = (S + I + D)/(C + S + D) \quad (7)$$

where C is the number of the correct entity and intent labels; S stands for a substitution error for misidentifying intent/slot. Insertion error (I), and deletion error (D) are specific to the SF task denoting extra and missing slots respectively. Note that the SemER metric is biased towards SF errors since each intent classification error will only be counted as 1 substitution error. Therefore, to better evaluate IC performance, we also independently present the multi-class intent classification error using the micro-averaged F1 score.

TABLE III: Performance on MultiATIS++ Dataset

	Agno		*Agno_noCaR*		*LA_feature*		*LA_aux*		*LA_mh*	
	SemER*↓	IC F1*↑	SemER↓	IC F1↑	SemER↓	IC F1↑	SemER↓	IC F1↑	SemER↓	IC F1↑
EN	6.49%	**97.42**%	6.67%	96.14%	**6.43**%	97.38%	6.54%	97.05%	6.67%	97.31%
DE	8.33%	95.40%	8.42%	95.27%	8.27%	95.44%	8.37%	95.44%	**8.14**%	**96.52**%
FR	10.05%	**95.95**%	10.97%	95.41%	**10.02**%	95.89%	10.12%	95.93%	10.19%	95.72%
ES	14.51%	97.91%	15.01%	97.18%	**13.99**%	**97.98**%	14.63%	98.15%	14.55%	97.92%
JA	11.56%	91.42%	11.66%	91.27%	11.32%	91.40%	11.50%	91.30%	**10.98**%	**92.89**%
PT	12.37%	97.21%	12.58%	97.14%	**12.09**%	**97.28**%	12.39%	97.08%	12.33%	97.07%
HI	21.02%	91.82%	21.79%	90.08%	20.20%	92.16%	**19.82**%	**92.21**%	19.88%	92.15%

*Lower SemER and higher IC F1 indicate better performance.

TABLE IV: Relative Change on Commercial IVA Dataset[2]

	Agno		*Agno_noCaR*		*LA_feature*		*LA_aux*		*LA_mh*	
	SemER*↓	IC F1*↑	SemER↓	IC F1↑	SemER↓	IC F1↑	SemER↓	IC F1↑	SemER↓	IC F1↑
EN	X	**X***	+0.79%	-0.66%	+1.52%	-0.33%	**-0.03**%	-0.02%	+0.01%	-0.03%
DE	**X**	**X**	+1.12%	-1.03%	+1.18%	-0.04%	+0.17%	+0.09%	+0.28%	-0.35%
FR	**X**	X	+0.18%	-0.09%	+0.17%	**+0.02**%	+0.24%	-1.37%	+0.71%	-0.15%
ES	X	X	+0.97%	-0.82%	**-1.20**%	**+0.22**%	-0.05%	+0.02%	-0.04%	-0.13%

*X stands for baseline performance. It is marked in bold if baseline performs the best.

TABLE V: Controlled Experiment on MultiATIS++ Dataset with Germanic/Romance Languages

	Agno		*Agno_noCaR*		*LA_feature*		*LA_aux*		*LA_mh*	
	SemER*↓	IC F1*↑	SemER↓	IC F1↑	SemER↓	IC F1↑	SemER↓	IC F1↑	SemER↓	IC F1↑
EN	6.40%	**97.44**%	6.58%	96.38%	**6.38**%	97.42%	6.52%	97.15%	6.61%	97.33%
DE	8.21%	95.62%	8.40%	95.37%	8.23%	95.48%	8.33%	95.52%	**8.19**%	**96.53**%
FR	**9.82**%	**96.01**%	10.43%	95.89%	9.93%	95.97%	10.10%	95.94%	10.23%	95.75%
ES	14.19%	**98.06**%	14.88%	97.26%	**13.97**%	98.03%	14.52%	98.17%	14.42%	97.90%

Training. The training for all multilingual NLU candidates was performed by finetuning the pretrained and distilled mBERT encoder. In the finetuning process for all candidate models, the word and position embeddings are frozen and the encoder layers are gradually unfrozen according to learning rate multipliers ranging from 0.01 to 0.1. Encoder representations are fed into two dense layer networks to predict intent and slot labels, each with one hidden layer of dimension 128 and 256 respectively. For both IC and SF decoders, GELU was used as the hidden activation with a dropout rate of 0.2 prior to the softmax/CRF layer. We use the Adam optimizer [28] with a learning rate of 1×10^{-4} and a batch size of 64. We implement early stopping based on the validation metric SemER with a maximum of 100 epochs. λ^C, and λ^{LT} are set as 1 and 0.8 respectively. These two hyperparameters were tuned by performing grid search and validating using the Dev set. All models were trained on a single NVIDIA TESLA V100 GPU.

B. Q1: Overall Multilingual IC-SF Task Performance

First, we present the overall IC-SF task performance benchmark using the SemER and IC F1 metrics described above on the public MultiATIS++ dataset (Table III). We can observe that language-aware candidates outperform $Agno$ in 5 out of 7 languages, indicating the effectiveness in incorporating language-specific capacities into the multilingual NLU modeling. Among the language-aware candidates, $LA\_feature$ and LA_mh consistently outperform LA_aux, except for Hindi. This observation may be correlated to the fact that LA_aux is trained to jointly optimize four tasks at the same time (i.e., IC, SF, LT, and CaR), thus subjecting to under fitting with the current distilled mBERT encoder. We thus recommend using a pretrained encoder with larger model capacity for LA_aux. We also observed that the CaR module did improve both IC F1 and SemER scores when comparing with $Agno$ and $Agno\_noCaR$ as an ablation study.

Next, we report the relative change on the IC-SF task performance benchmark using the commercial VA dataset (Table IV). Again, we found consistent improvement in all languages when introducing the CaR module. Surprisingly, we found that only $LA\_feature$ outperforms $Agno$ in FR and ES by a very slight margin, whereas the language-agnostic approach achieves on par or better performance in most metrics.

We attribute such discrepancy to the language dissimilarity of the datasets. In particular, MultiATIS++ includes a diverse family of languages, including Germanic/Romance (EN, DE, FR, PT, ES), Indo-Euro (Hindi), and Japonic language (Japanese) that are linguistically dissimilar [29]. Therefore, language awareness can benefit most by allowing the model to explicitly consider language differences. In contrast, the commercial VA dataset only contains Romance/Germanic languages that are linguistically similar, where a joint encoder/decoder (i.e.,

[2]Due to anonymity and legal requirements for the commercial IVA dataset, we only report relative SemER and IC F1 gains. For the public dataset MultiATIS++, we report absolute performance metrics.

TABLE VI: Relative Change on Code-Mixed Commercial IVA Data

	Agno		***LA_feature***		***LA_aux***		***LA_mh***	
	SemER↓	IC F1↑	SemER↓	IC F1↑	SemER↓	IC F1↑	SemER↓	IC F1↑
DE	**X**	**X**	+2.17%	-0.51%	+0.54%	-0.17%	+4.72%	-2.40%
FR	X	**X**	+3.43%	-1.45%	**-1.32%**	-0.02%	+4.12%	-1.95%
ES	X	X	+2.92%	-1.90%	**-1.63%**	**+0.59%**	+2.43%	-0.83%

language-aware) design can be more beneficial. To further validate this conclusion, we perform a controlled experiment where we intentionally pick the same linguistically similar languages in int MultiATIS++ dataset as those in the Commercial IVA dataset, namely EN, DE, FR, and ES. The results are summarized in Table V. We can make two observations. First, by removing dissimilar languages, the overall performance for all candidates and all locales got improved. Second, the advantage of language-aware candidates becomes less significant when dissimilar languages are removed. For example, LA_mh only outperforms $Agno$ in DE by 0.2% as compared to the previous performance gain of 2.3% without removing these languages. Both observations echo the findings in [12] and [20] that mixing languages with a dissimilar alphabet and linguistic features may degrade the effectiveness of multilingual language representation on downstream tasks.

C. Q2: Performance with Code-Mixed Input

In this set of experiments, we study how each model handles code-mixed user queries. In a practical IVA system, it is common for users to provide code-mixed queries where language code for the input utterance is either non-available (i.e., not labeled by user) or ambiguous.

To emulate a real-world IVA system, we first perform the language identification (LID) task for the input utterances using FastText LID [30]. Then we use the detected language code as input to $LA\_feature$, and LA_mh. For $Agno$ and LA_aux, we skip the LID procedure since both require no language code information during inference time.

In this experiment, we randomly sampled an additional evaluation set of 6,000 code-mixed utterances for DE, FR, and ES from the real-world commercial VA traffic. These utterances are code-mixed with English. We skipped the MultiATIS++ dataset since it does not contain code-mixed utterances. The results are summarized in Table VI. We can observe that LA_aux is on par or better compared to $Agno$ and significantly outperforms LA_aux and LA_mh for code-mixed utterances. We attribute this performance gain to its token-level language tagger, which allows the model to better determine the slot label for each token based on its detected language. In contrast, $LA\_feature$ and LA_mh depend on accurate utterance-level language code. Therefore, they cannot handle utterances with tokens from multiple languages.

D. Q3: Training Time and Model Sizes

Finally, we explore the training time and model size trade-offs among the proposed models. In this experiment, we chose to use the MultiATIS++ dataset and vary the number of languages for training from 1 (monolingual) to 7. The training time is evaluated by averaging per-epoch training time, and the model size is represented by the number of parameters of the proposed models. The results for both training time and model sizes are reported as relative increase compared to the language-agnostic baseline - Agno. The results are summarized in Figure 2-5. We can observe that all language-aware models increased the training time compared to the language-agnostic model. In particular, LA_mh has the largest increase in training time at 37.9%, 62.1%, and 98.9% increase when the number of languages is 3, 5, 7 respectively. These increases are due to the extra numbers of decoders to be trained, especially when the number of languages is large. $LA\_feature$ results in trivial change (<3% increase) in training time.

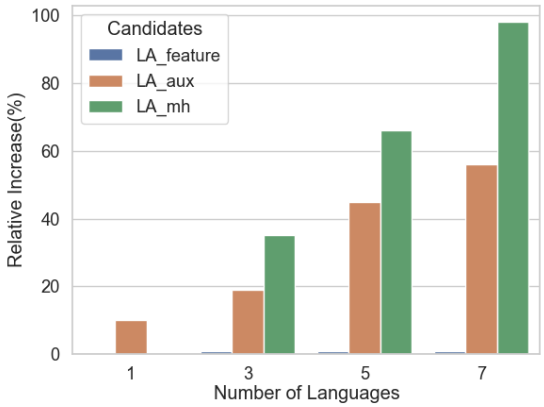

Fig. 6: Training Time Compared to $Agno$

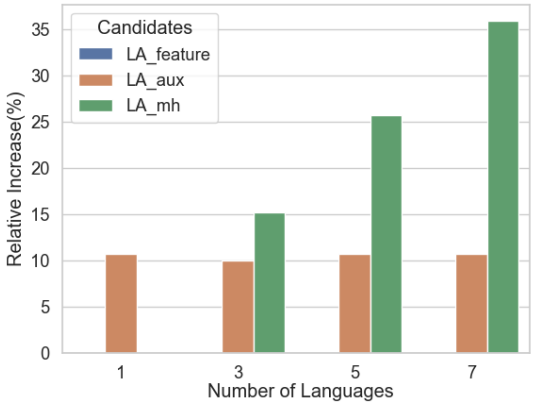

Fig. 7: Model Parameters Compared to $Agno$

Similarly, it is observed that LA_mh leads to the largest increase in terms of model sizes (+35.0%) when training all 7 languages, due to the stacking of individual decoders per

language. In contrast, LA_aux introduces a 10.78% increase in model size, such an increase is invariant of the number of languages to be trained; while $LA\_feature$ leads to no change in model size at all. Both findings in training time and model size suggest LA_mh is less scalable compared to all other candidates when applying to a large number of languages.

VII. Conclusion

In this paper, we make the first effort to explore language-agnostic and language-aware designs for multilingual NLU and perform systematic comparisons about their trade-offs. A real-world large-scale commercial IVA system, as well as a public NLU dataset with a diverse set of languages, are used to validate the proposed approaches. We have found that language-aware multilingual NLU designs are useful when the language compositions are diversified and dissimilar. Surprisingly, a simple language code as feature design ($LA\_feature$) is able to achieve the best performance compared to other language-aware designs in many scenarios. We also found that explicitly adding token-level language identification as an auxiliary task benefits the NLU performance on the code-mixed dataset. Finally, we found adding extra model capacity in the decoding process (i.e., LA_mh) did not improve the NLU performance significantly compared to other designs with less model capacity, while at the cost of extra model size and training time. These findings provide valuable insights into multilingual NLU modeling. We leave a few directions as future work. We plan to explore other pretrained multilingual encoders such as XLM-RoBERTa [31] to further generalize our findings. We will also explore cross-lingual transfer settings and compare how language-agnostic and language-aware approaches perform in low-resource languages.

References

[1] R. Sarikaya, G. E. Hinton, and A. Deoras, "Application of deep belief networks for natural language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 778–784, 2014.

[2] L. Dong, N. Yang, W. Wang, F. Wei, X. Liu, Y. Wang, J. Gao, M. Zhou, and H.-W. Hon, "Unified language model pre-training for natural language understanding and generation," *arXiv preprint arXiv:1905.03197*, 2019.

[3] A. Jaech, L. Heck, and M. Ostendorf, "Domain adaptation of recurrent neural networks for natural language understanding," *arXiv preprint arXiv:1604.00117*, 2016.

[4] J. Lee, R. Sarikaya, and Y.-B. Kim, "Locale-agnostic universal domain classification model in spoken language understanding," *arXiv preprint arXiv:1905.00924*, 2019.

[5] J. Cao, J. Wang, W. Hamza, K. Vanee, and S.-W. Li, "Style attuned pre-training and parameter efficient fine-tuning for spoken language understanding," *arXiv preprint arXiv:2010.04355*, 2020.

[6] Y.-B. Kim, S. Lee, and K. Stratos, "Onenet: Joint domain, intent, slot prediction for spoken language understanding," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 547–553.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[8] Q. Li, "Literature survey: domain adaptation algorithms for natural language processing," *Department of Computer Science The Graduate Center, The City University of New York*, pp. 8–10, 2012.

[9] W. Xu, B. Haider, and S. Mansour, "End-to-end slot alignment and recognition for cross-lingual nlu," *arXiv preprint arXiv:2004.14353*, 2020.

[10] Q. N. T. Do and J. Gaspers, "Cross-lingual transfer learning for spoken language understanding," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5956–5960.

[11] B. Zhang, A. Bapna, R. Sennrich, and O. Firat, "Share or not? learning to schedule language-specific capacity for multilingual translation," in *International Conference on Learning Representations*, 2020.

[12] Z. Yang, R. Salakhutdinov, and W. W. Cohen, "Transfer learning for sequence tagging with hierarchical recurrent networks," *arXiv preprint arXiv:1703.06345*, 2017.

[13] S. Sen, K. K. Gupta, A. Ekbal, and P. Bhattacharyya, "Multilingual unsupervised nmt using shared encoder and language-specific decoders," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 3083–3089.

[14] K. M. Sathyendra, S. Choudhary, and L. Nicolich-Henkin, "Extreme model compression for on-device natural language understanding," *arXiv preprint arXiv:2012.00124*, 2020.

[15] S. Sridhar and M. E. Tolentino, "Evaluating voice interaction pipelines at the edge," in *2017 IEEE International Conference on Edge Computing (EDGE)*. IEEE, 2017, pp. 248–251.

[16] Y. Gao, J. Feng, Y. Liu, L. Hou, X. Pan, and Y. Ma, "Code-switching sentence generation by bert and generative adversarial networks." in *INTERSPEECH*, 2019, pp. 3525–3529.

[17] B. Liu and I. Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling," *arXiv preprint arXiv:1609.01454*, 2016.

[18] Q. Chen, Z. Zhuo, and W. Wang, "Bert for joint intent classification and slot filling," *arXiv preprint arXiv:1902.10909*, 2019.

[19] G. Castellucci, V. Bellomaria, A. Favalli, and R. Romagnoli, "Multi-lingual intent detection and slot filling in a joint bert-based model," *arXiv preprint arXiv:1907.02884*, 2019.

[20] Y. Lin, S. Yang, V. Stoyanov, and H. Ji, "A multi-lingual multi-task architecture for low-resource sequence labeling," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 799–809.

[21] B. Samanta, S. Reddy, H. Jagirdar, N. Ganguly, and S. Chakrabarti, "A deep generative model for code-switched text," *arXiv preprint arXiv:1906.08972*, 2019.

[22] T. Pires, E. Schlinger, and D. Garrette, "How multilingual is multilingual bert?" *arXiv preprint arXiv:1906.01502*, 2019.

[23] S. Santy, A. Srinivasan, and M. Choudhury, "Bertologicomix: How does code-mixing interact with multilingual bert?" in *Proceedings of the Second Workshop on Domain Adaptation for NLP*, 2021, pp. 111–121.

[24] R. De Mori, F. Bechet, D. Hakkani-Tur, M. McTear, G. Riccardi, and G. Tur, "Spoken language understanding," *IEEE Signal Processing Magazine*, vol. 25, no. 3, pp. 50–58, 2008.

[25] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

[26] U. Barman, A. Das, J. Wagner, and J. Foster, "Code mixing: A challenge for language identification in the language of social media," in *Proceedings of the first workshop on computational approaches to code switching*, 2014, pp. 13–23.

[27] J. Makhoul, F. Kubala, R. Schwartz, R. Weischedel *et al.*, "Performance measures for information extraction," in *Proceedings of DARPA broadcast news workshop*. Herndon, VA, 1999, pp. 249–252.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[29] A. Siddhant, M. Johnson, H. Tsai, N. Ari, J. Riesa, A. Bapna, O. Firat, and K. Raman, "Evaluating the cross-lingual effectiveness of massively multilingual neural machine translation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 8854–8861.

[30] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, "Fasttext.zip: Compressing text classification models," *arXiv preprint arXiv:1612.03651*, 2016.

[31] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," *arXiv preprint arXiv:1911.02116*, 2019.