Natural Language Interactions in Autonomous Vehicles: Intent Detection and Slot Filling from Passenger Utterances

Eda Okur, Shachi H Kumar, Saurav Sahay, Asli Arslan Esme, and Lama Nachman

Intel Labs, Anticipatory Computing Lab, USA
{eda.okur, shachi.h.kumar, saurav.sahay, asli.arslan.esme, lama.nachman}@intel.com

Abstract. Understanding passenger intents and extracting relevant slots are important building blocks towards developing contextual dialogue systems for natural interactions in autonomous vehicles (AV). In this work, we explored AMIE (Automated-vehicle Multi-modal In-cabin Experience), the in-cabin agent responsible for handling certain passenger-vehicle interactions. When the passengers give instructions to AMIE, the agent should parse such commands properly and trigger the appropriate functionality of the AV system. In our current explorations, we focused on AMIE scenarios describing usages around setting or changing the destination and route, updating driving behavior or speed, finishing the trip and other use-cases to support various natural commands. We collected a multi-modal in-cabin dataset with multi-turn dialogues between the passengers and AMIE using a Wizard-of-Oz scheme via a realistic scavenger hunt game activity. After exploring various recent Recurrent Neural Networks (RNN) based techniques, we introduced our own hierarchical joint models to recognize passenger intents along with relevant slots associated with the action to be performed in AV scenarios. Our experimental results outperformed certain competitive baselines and achieved overall F1-scores of 0.91 for utterance-level intent detection and 0.96 for slot filling tasks. In addition, we conducted initial speech-to-text explorations by comparing intent/slot models trained and tested on human transcriptions versus noisy Automatic Speech Recognition (ASR) outputs. Finally, we compared the results with single passenger rides versus the rides with multiple passengers.

Keywords: Intent recognition · Slot filling · Hierarchical joint learning · Spoken language understanding (SLU) · In-cabin dialogue agent.

1 Introduction

One of the exciting yet challenging areas of research in Intelligent Transportation Systems is developing context-awareness technologies that can enable autonomous vehicles to interact with their passengers, understand passenger con-

text and situations, and take appropriate actions accordingly. To this end, building multi-modal dialogue understanding capabilities situated in the in-cabin context is crucial to enhance passenger comfort and gain user confidence in AV interaction systems. Among many components of such systems, intent recognition and slot filling modules are one of the core building blocks towards carrying out successful dialogue with passengers. As an initial attempt to tackle some of those challenges, this study introduce in-cabin intent detection and slot filling models to identify passengers' intent and extract semantic frames from the natural language utterances in AV. The proposed models are developed by leveraging User Experience (UX) grounded realistic (ecologically valid) in-cabin dataset. This dataset is generated with naturalistic passenger behaviors, multiple passenger interactions, and with presence of a Wizard-of-Oz (WoZ) agent in moving vehicles with noisy road conditions.

1.1 Background

Long Short-Term Memory (LSTM) networks [7] are widely-used for temporal sequence learning or time-series modeling in Natural Language Processing (NLP). These neural networks are commonly employed for sequence-to-sequence (seq2seq) and sequence-to-one (seq2one) modeling problems, including slot filling tasks [11] and utterance-level intent classification [5, 16] which are well-studied for various application domains. Bidirectional LSTMs (Bi-LSTMs) [17] are extensions of traditional LSTMs which are proposed to improve model performance on sequence classification problems even further. Jointly modeling slot extraction and intent recognition [5, 24] is also explored in several architectures for task-specific applications in NLP. Using Attention mechanism [15, 23] on top of RNNs is yet another recent break-through to elevate the model performance by attending inherently crucial sub-modules of given input. There exist various architectures to build hierarchical learning models [26, 10, 21] for document-to-sentence level, and sentence-to-word level classification tasks, which are highly domain-dependent and task-specific.

Automatic Speech Recognition (ASR) technology has recently achieved human-level accuracy in many fields [22, 19]. For spoken language understanding (SLU), it is shown that training SLU models on true text input (i.e., human transcriptions) versus noisy speech input (i.e., ASR outputs) can achieve varying results [9]. Even greater performance degradations are expected in more challenging and realistic setups with noisy environments, such as moving vehicles in actual traffic conditions. As an example, a recent work [25] attempts to classify sentences as navigation-related or not using the DARPA supported CU-Move in-vehicle speech corpus [6], a relatively old and large corpus focusing on route navigation. For this binary intent classification task, the authors observed that detection performances are largely affected by high ASR error rates due to background noise and multi-speakers in CU-Move dataset (not publicly available). For in-cabin dialogue between car assistants and driver/passengers, recent studies explored creating a public dataset using a WoZ approach [3], and improving ASR for passenger speech recognition [4].

A preliminary report on research designed to collect data for human-agent interactions in a moving vehicle is presented in a previous study [18], with qualitative analysis on initial observations and user interviews. Our current study is focused on the quantitative analysis of natural language interactions found in this in-vehicle dataset, where we address intent detection and slot extraction tasks for passengers interacting with the AMIE in-cabin agent.

Contributions. In this study, we propose a UX grounded realistic intent recognition and slot filling models with naturalistic passenger-vehicle interactions in moving vehicles. Based on observed interactions, we defined in-vehicle intent types and refined their relevant slots through a data driven process. After exploring existing approaches for jointly training intents and slots, we applied certain variations of these models that perform best on our dataset to support various natural commands for interacting with the car-agent. The main differences in our proposed models can be summarized as follows: (1) Using the extracted intent keywords in addition to the slots to jointly model them with utterance-level intents (where most of the previous work [26, 10] only join slots and utterance-level intents, ignoring the intent keywords); (2) The 2-level hierarchy we defined by word-level detection/extraction for slots and intent keywords first, and then filtering-out predicted non-slot and non-intent keywords instead of feeding them into the upper levels of the network (i.e., instead of using stacked RNNs with multiple recurrent hidden layers for the full utterance [10, 21], which are computationally costly for long utterances with many non-slot & non-intent-related words), and finally using only the predicted valid-slots and intent-related keywords as an input to the second level of the hierarchy; (3) Extending joint models [5, 24] to include both beginning-of-utterance and end-of-utterance tokens to leverage Bi-LSTMs (after observing that we achieved better results by doing so). We compared our intent detection and slot filling results with the results obtained from Dialogflow[1], a commercially available intent-based dialogue system by Google, and showed that our proposed models perform better for both tasks on the same dataset. We also conducted initial speech-to-text explorations by comparing models trained and tested (10-fold CV) on human transcriptions versus noisy ASR outputs (via Cloud Speech-to-Text[2]). Finally, we compared the results with single passenger rides versus the rides with multiple passengers.

2 Methodology

2.1 Data Collection and Annotation

Our AV in-cabin dataset includes around 30 hours of multi-modal data collected from 30 passengers (15 female, 15 male) in a total of 20 rides/sessions. In 10 sessions, single passenger was present (i.e., *singletons*), whereas the remaining

[1] https://dialogflow.com
[2] https://cloud.google.com/speech-to-text/

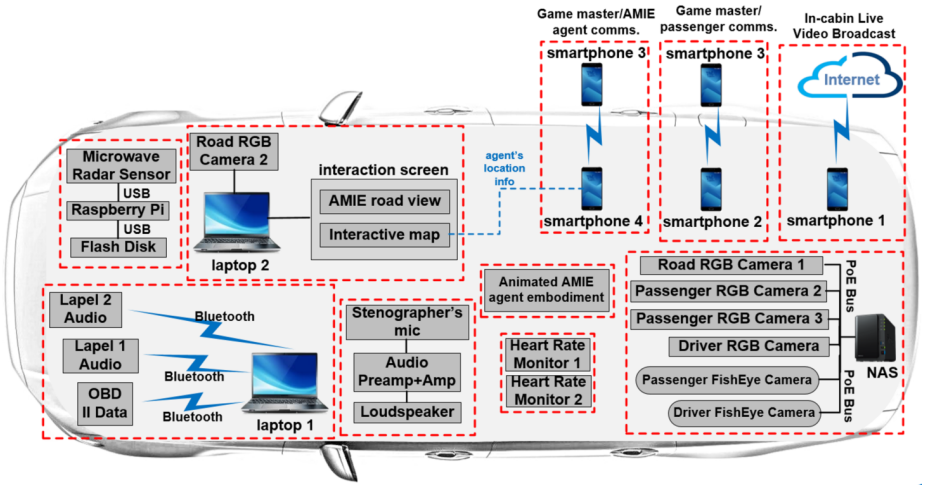


Fig. 1: AMIE In-cabin Data Collection Setup

10 sessions include two passengers (i.e., *dyads*) interacting with the vehicle. The data is collected "in the wild" on the streets of Richmond, British Columbia, Canada. Each ride lasted about 1 hour or more. The vehicle is modified to hide the operator and the human acting as in-cabin agent from the passengers, using a variation of WoZ approach [20]. Participants sit in the back of the car and are separated by a semi-sound proof and translucent screen from the human driver and the WoZ AMIE agent at the front. In each session, the participants were playing a scavenger hunt game by receiving instructions over the phone from the Game Master. Passengers treat the car as AV and communicate with the WoZ AMIE agent via speech commands. Game objectives require passengers to interact naturally with the agent to go to certain destinations, update routes, stop the vehicle, give specific directions regarding where to pull over or park (sometimes with gesture), find landmarks, change speed, get in and out of the vehicle, etc. Further details of the data collection design and scavenger hunt protocol can be found in the preliminary study [18]. See Fig. 1 for the vehicle instrumentation to enhance multi-modal data collection setup. Our study is the initial work on this multi-modal dataset to develop intent detection and slot filling models, where we leveraged from the back-driver video/audio stream recorded by an RGB camera (facing towards the passengers) for manual transcription and annotation of in-cabin utterances. In addition, we used the audio data recorded by Lapel 1 Audio and Lapel 2 Audio (Fig. 1) as our input resources for the ASR.

For in-cabin intent understanding, we described 4 groups of usages to support various natural commands for interacting with the vehicle: (1) *Set/Change Destination/Route* (including turn-by-turn instructions), (2) *Set/Change Driving Behavior/Speed*, (3) *Finishing the Trip Use-cases*, and (4) *Others* (open/close door/window/trunk, turn music/radio on/off, change AC/temperature, show map, etc.). According to those scenarios, 10 types of passenger intents are iden-

Table 1: AMIE Dataset Statistics: Utterance-level Intent Types

AMIE Scenario	**Intent Type**	**Utterance Count**
Finishing the Trip Use-cases	Stop	317
	Park	450
	PullOver	295
	DropOff	281
Set/Change Destination/Route	SetDestination	552
	SetRoute	676
Set/Change Driving Behavior/Speed	GoFaster	265
	GoSlower	238
Others (Door, Music, etc.)	OpenDoor	142
	Other	202
	Total	*3418*

tified and annotated as follows: *SetDestination*, *SetRoute*, *GoFaster*, *GoSlower*, *Stop*, *Park*, *PullOver*, *DropOff*, *OpenDoor*, and *Other*. For slot filling task, relevant slots are identified and annotated as: *Location*, *Position/Direction*, *Object*, *Time Guidance*, *Person*, *Gesture/Gaze* (e.g., 'this', 'that', 'over there', etc.), and *None/O*. In addition to utterance-level intents and slots, word-level intent related keywords are annotated as *Intent*. We obtained 1331 utterances having commands to AMIE agent from our in-cabin dataset. We expanded this dataset via the creation of similar tasks on Amazon Mechanical Turk [2] and reached 3418 utterances with intents in total. Intent and slot annotations are obtained on the transcribed utterances by majority voting of 3 annotators. Those annotation results for utterance-level intent types, slots and intent keywords can be found in Table 1 and Table 2 as a summary of dataset statistics.

Table 2: AMIE Dataset Statistics: Slots and Intent Keywords

Slot Type	**Slot Count**	**Keyword Type**	**Keyword Count**
Location	4460	Intent	5921
Position/Direction	3187	Non-Intent	25000
Person	1360	Valid-Slot	10954
Object	632	Non-Slot	19967
Time Guidance	792	Intent $\cup$ Valid-Slot	16875
Gesture/Gaze	523	Non-Intent $\cap$ Non-Slot	14046
None	19967		
Total	*30921*	*Total*	*30921*

2.2 Detecting Utterance-level Intent Types

As a baseline system, we implemented term-frequency and rule-based mapping mechanisms from word-level intent keywords extraction to utterance-level intent recognition. To further improve the utterance-level performance, we explored various RNN architectures and developed a hierarchical (2-level) models to recognize passenger intents along with relevant entities/slots in utterances. Our hierarchical model has the following 2-levels:

- Level-1: Word-level extraction (to automatically detect/predict and eliminate non-slot & non-intent keywords first, as they would not carry much information for understanding the utterance-level intent-type).
- Level-2: Utterance-level recognition (to detect final intent-types from given utterances, by using valid slots and intent keywords as inputs only, which are detected at Level-1).

RNN with LSTM Cells for Sequence Modeling. In this study, we employed an RNN architecture with LSTM cells that are designed to exploit long range dependencies in sequential data. LSTM has memory cell state to store relevant information and various gates, which can mitigate the vanishing gradient problem [7]. Given the input x_t at time t, and hidden state from the previous time step h_{t-1}, the hidden and output layers for the current time step are computed. The LSTM architecture is specified by the following equations:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{1}$$
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{2}$$
$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{3}$$
$$g_t = \tanh(W_{xg}x_t + W_{hg}h_{t-1} + b_g) \tag{4}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{5}$$
$$h_t = o_t \odot \tanh(c_t) \tag{6}$$

where W and b denote the weight matrices and bias terms, respectively. The sigmoid (σ) and tanh are activation functions applied element-wise, and $\odot$ denotes the element-wise vector product. LSTM has a memory vector c_t to read/write or reset using a gating mechanism and activation functions. Here, input gate i_t scales down the input, the forget gate f_t scales down the memory vector c_t, and the output gate o_t scales down the output to achieve final h_t, which is used to predict y_t (through a $softmax$ activation). Similar to LSTMs, GRUs [1] are proposed as a simpler and faster alternative, having only reset and update gates. For Bi-LSTM [17, 5], two LSTM architectures are traversed in forward and backward directions, where their hidden layers are concatenated to compute the output.

Extracting Slots and Intent Keywords. For slot filling and intent keywords extraction, we experimented with various configurations of seq2seq LSTMs [16] and GRUs [1], as well as Bi-LSTMs [17]. A sample network architecture can be

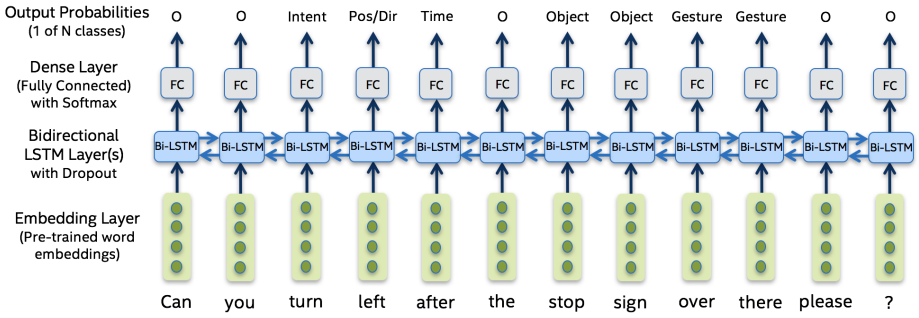


Fig. 2: Seq2seq Bi-LSTM Network for Slot Filling and Intent Keyword Extraction

seen in Fig. 2 where we jointly trained slots and intent keywords. The passenger utterance is fed into LSTM/GRU network via an embedding layer as a sequence of words, which are transformed into word vectors. We also experimented with GloVe [14], word2vec [12, 13], and fastText [8] as pre-trained word embeddings. To prevent overfitting, we used a dropout layer with 0.5 rate for regularization. Best performing results are obtained with Bi-LSTMs and GloVe embeddings (6B tokens, 400K vocabulary size, vector dimension 100).

Utterance-level Recognition. For utterance-level intent detection, we mainly experimented with 5 groups of models: (1) Hybrid: RNN + Rule-based, (2) Separate: Seq2one Bi-LSTM with Attention, (3) Joint: Seq2seq Bi-LSTM for slots/intent keywords & utterance-level intents, (4) Hierarchical & Separate, (5) Hierarchical & Joint. For (1), we detect/extract intent keywords and slots (via RNN) and map them into utterance-level intent-types (rule-based). For (2), we feed the whole utterance as input sequence and intent-type as single target into Bi-LSTM network with Attention mechanism. For (3), we jointly train word-level intent keywords/slots and utterance-level intents (by adding $<BOU>/<EOU>$ terms to the beginning/end of utterances with intent-types as their labels). For (4) and (5), we detect/extract intent keywords/slots first, and then only feed the predicted keywords/slots as a sequence into (2) and (3), respectively.

3 Experiments and Results

3.1 Utterance-Level Intent Detection Experiments

The details of 5 groups of models and their variations that we experimented with for utterance-level intent recognition are summarized in this section.

Hybrid Models. Instead of purely relying on machine learning (ML) or deep learning (DL) system, hybrid models leverage both ML/DL and rule-based systems. In this model, we defined our hybrid approach as using RNNs first for

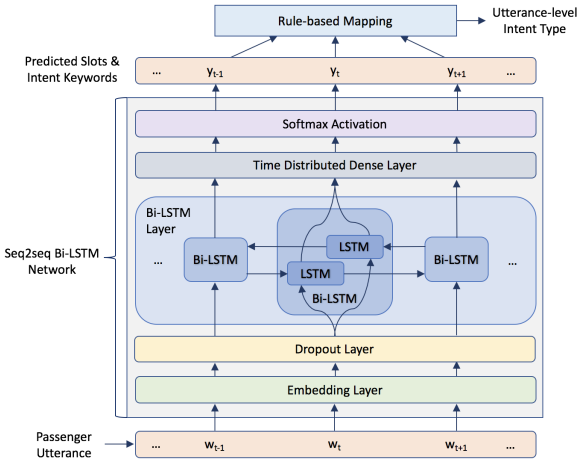
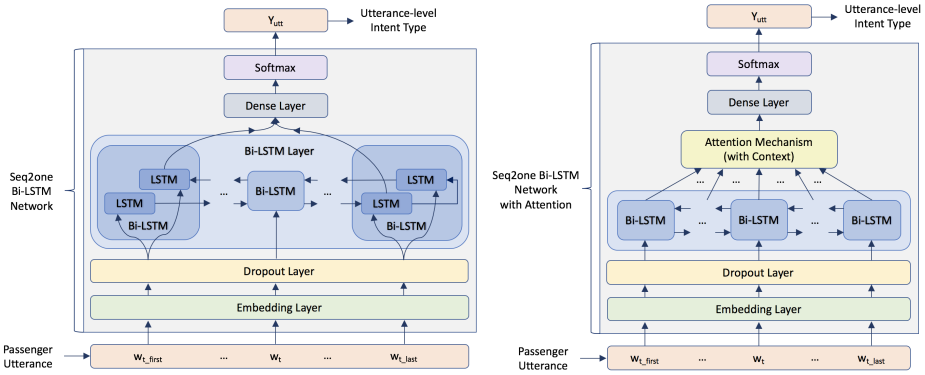


Fig. 3: Hybrid Models Network Architecture

detecting/extracting *intent keywords* and *slots*; then applying rule-based mapping mechanisms to identify *utterance-level intents* (using the predicted *intent keywords* and *slots*). A sample network architecture can be seen in Fig. 3 where we leveraged seq2seq Bi-LSTM networks for word-level extraction before the rule-based mapping to utterance-level intent classes. The model variations are defined based on varying mapping mechanisms and networks as follows:

- Hybrid-0: RNN (Seq2seq LSTM for *intent keywords* extraction) + Rule-based (mapping extracted *intent keywords* to *utterance-level intents*)
- Hybrid-1: RNN (Seq2seq Bi-LSTM for *intent keywords* extraction) + Rule-based (mapping extracted *intent keywords* to *utterance-level intents*)
- Hybrid-2: RNN (Seq2seq Bi-LSTM for *intent keywords* & *slots* extraction) + Rule-based (mapping extracted *intent keywords* & *Position/Direction slots* to *utterance-level intents*)
- Hybrid-3: RNN (Seq2seq Bi-LSTM for *intent keywords* & *slots* extraction) + Rule-based (mapping extracted *intent keywords* & *all slots* to *utterance-level intents*)

Separate Seq2one Models. This approach is based on separately training sequence-to-one RNNs for *utterance-level intents* only. These are called separate models as we do not leverage any information from the *slot* or *intent keyword* tags (i.e., *utterance-level intents* are not jointly trained with *slots*/*intent keywords*). Note that in seq2one models, we feed the utterance as an input sequence and the LSTM layer will only return the hidden state output at the last time step. This single output (or concatenated output of last hidden states from the forward and backward LSTMs in Bi-LSTM case) will be used to classify the intent type of the given utterance. The idea behind is that the last hidden state of



(a) Separate Seq2one Network (b) Separate Seq2one with Attention

Fig. 4: Separate Models Network Architecture

the sequence will contain a latent semantic representation of the whole input utterance, which can be utilized for utterance-level intent prediction. See Fig. 4 (a) for sample network architecture of the seq2one Bi-LSTM network. Note that in the Bi-LSTM implementation for seq2one learning (i.e., when not returning sequences), the outputs of backward/reverse LSTM is actually ordered in reverse time steps (t_{last} ... t_{first}). Thus, as illustrated in Fig. 4 (a), we actually concatenate the hidden state outputs of forward LSTM at last time step and backward LSTM at first time step (i.e., first word in a given utterance), and then feed this merged result to the dense layer. Fig. 4 (b) depicts the seq2one Bi-LSTM network with Attention mechanism applied on top of Bi-LSTM layers. For the Attention case, the hidden state outputs of all time steps are fed into the Attention mechanism that will allow to point at specific words in a sequence when computing a single output [15]. Another variation of Attention mechanism we examined is the AttentionWithContext, which incorporates a context/query vector jointly learned during the training process to assist the attention [23]. All seq2one model variations we experimented with can be summarized as follows:

- Separate-0: Seq2one LSTM for *utterance-level intents*
- Separate-1: Seq2one Bi-LSTM for *utterance-level intents*
- Separate-2: Seq2one Bi-LSTM with Attention [15] for *utterance-level intents*
- Separate-3: Seq2one Bi-LSTM with AttentionWithContext [23] for *utterance-level intents*

Joint Seq2seq Models. Using sequence-to-sequence networks, the approach here is jointly training annotated *utterance-level intents* and *slots/intent keywords* by adding *<BOU>/ <EOU>* tokens to the beginning/end of each utterance, with *utterance-level intent-type* as labels of such tokens. Our approach is an extension of [5], in which only an *<EOS>* term is added with *intent-type*

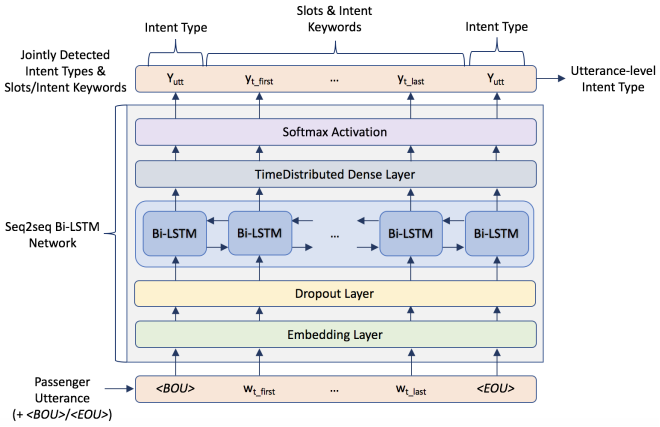
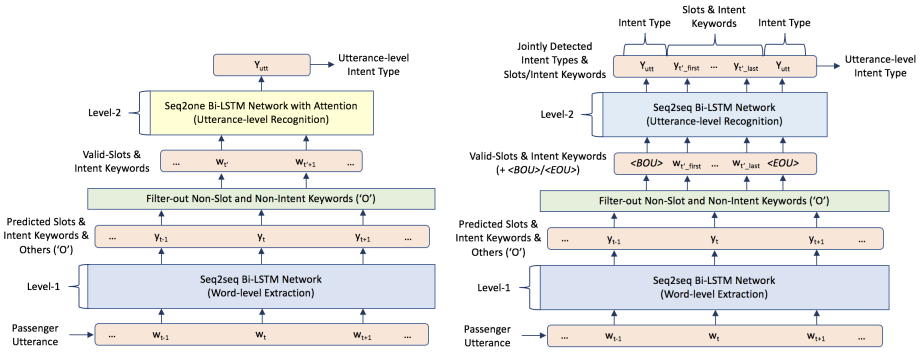


Fig. 5: Joint Models Network Architecture

tags associated to this sentence final token, both for LSTM and Bi-LSTM cases. However, we experimented with adding both $<BOU>$ and $<EOU>$ terms as Bi-LSTMs will be used for seq2seq learning, and we observed that slightly better results can be achieved by doing so. The idea behind is that, since this is a seq2seq learning problem, at the last time step (i.e., prediction at $<EOU>$) the reverse pass in Bi-LSTM would be incomplete (refer to Fig. 4 (a) to observe the last Bi-LSTM cell). Therefore, adding $<BOU>$ token and leveraging the backward LSTM output at first time step (i.e., prediction at $<BOU>$) would potentially help for joint seq2seq learning. An overall network architecture can be found in Fig. 5 for our joint models. We will report the experimental results on two variations (with and without *intent keywords*) as follows:

- Joint-1: Seq2seq Bi-LSTM for *utterance-level intent* detection (jointly trained with *slots*)
- Joint-2: Seq2seq Bi-LSTM for *utterance-level intent* detection (jointly trained with *slots* & *intent keywords*)

Hierarchical & Separate Models. Proposed hierarchical models are detecting/extracting *intent keywords* & *slots* using sequence-to-sequence networks first (i.e., level-1), and then feeding only the words that are predicted as *intent keywords* & *valid slots* (i.e., not the ones that are predicted as *None/O*) as an input sequence to various separate sequence-to-one models (described above) to recognize final *utterance-level intents* (i.e., level-2). A sample network architecture is given in Fig. 6 (a). The idea behind filtering out non-slot and non-intent keywords here resembles providing a summary of input sequence to the upper levels of the network hierarchy, where we actually learn this summarized sequence of keywords using another RNN layer. This would potentially result in focusing the utterance-level classification problem on the most salient words of the input sequences (i.e., *intent keywords* & *slots*) and also effectively reducing the



(a) Hierarchical & Separate Model (b) Hierarchical & Joint Model

Fig. 6: Hierarchical Models Network Architecture

length of input sequences (i.e., improving the long-term dependency issues observed in longer sequences). Note that according to our dataset statistics given in Table 2, 45% of the words found in transcribed utterances with passenger intents are annotated as non-slot and non-intent keywords (e.g., 'please', 'okay', 'can', 'could', incomplete/interrupted words, filler sounds like 'uh'/'um', certain stop words, punctuation, and many others that are not related to intent/slots). Therefore, the proposed approach would result in reducing the sequence length nearly by half at the input layer of level-2 for utterance-level recognition. For hierarchical & separate models, we experimented with 4 variations based on which separate model used at the second level of the hierarchy, and these are summarized as follows:

- Hierarchical & Separate-0: Level-1 (Seq2seq LSTM for *intent keywords* & *slots* extraction) + Level-2 (Separate-0: Seq2one LSTM for *utterance-level intent* detection)
- Hierarchical & Separate-1: Level-1 (Seq2seq Bi-LSTM for *intent keywords* & *slots* extraction) + Level-2 (Separate-1: Seq2one Bi-LSTM for *utterance-level intent* detection)
- Hierarchical & Separate-2: Level-1 (Seq2seq Bi-LSTM for *intent keywords* & *slots* extraction) + Level-2 (Separate-2: Seq2one Bi-LSTM + Attention for *utterance-level intent* detection)
- Hierarchical & Separate-3: Level-1 (Seq2seq Bi-LSTM for *intent keywords* & *slots* extraction) + Level-2 (Separate-3: Seq2one Bi-LSTM + AttentionWithContext for *utterance-level intent* detection)

Hierarchical & Joint Models. Proposed hierarchical models detect/extract *intent keywords* & *slots* using sequence-to-sequence networks first, and then only the words that are predicted as *intent keywords* & *valid slots* (i.e., not the ones that are predicted as *None/O*) are fed as input to the joint sequence-to-sequence

models (described above). See Fig. 6 (b) for sample network architecture. After the filtering or summarization of sequence at level-1, <*BOU*> and <*EOU*> tokens are appended to the shorter input sequence before level-2 for joint learning. Note that in this case, using Joint-1 model (jointly training annotated *slots* & *utterance-level intents*) for the second level of the hierarchy would not make much sense (without *intent keywords*). Hence, Joint-2 model is used for the second level as described below:

– Hierarchical & Joint-2: Level-1 (Seq2seq Bi-LSTM for *intent keywords* & *slots* extraction) + Level-2 (Joint-2 Seq2seq models with *slots* & *intent keywords* & *utterance-level intents*)

Table 3 summarizes the results of various approaches we investigated for utterance-level intent understanding. We achieved 0.91 overall F1-score with our best-performing model, namely *Hierarchical & Joint-2*. All model results are obtained via 10-fold cross-validation (10-fold CV) on the same dataset. For our AMIE scenarios, Table 4 shows the intent-wise detection results with the initial (*Hybrid-0*) and currently best performing (*H-Joint-2*) intent recognizers. With our best model (*H-Joint-2*), relatively problematic *SetDestination* and *SetRoute* intents detection performances in baseline model (*Hybrid-0*) jumped from 0.78 to 0.89 and 0.75 to 0.88, respectively.

We compared our intent detection results with the Dialogflow's Detect Intent API. The same AMIE dataset is used to train and test (10-fold CV) Dialogflow's intent detection and slot filling modules, using the recommended hybrid mode (rule-based and ML). As shown in Table 4, an overall F1-score of 0.89 is achieved with Dialogflow for the same task. As you can see, our *Hierarchical & Joint* models obtained higher results than the Dialogflow for 8 out of 10 intent types.

Table 3: Utterance-level Intent Detection Performance Results (10-fold CV)

Model Type	***Prec***	***Rec***	**F1**
Hybrid-0: RNN (LSTM) + Rule-based (*intent keywords*)	*0.86*	*0.85*	0.85
Hybrid-1: RNN (Bi-LSTM) + Rule-based (*intent keywords*)	*0.87*	*0.86*	0.86
Hybrid-2: RNN (Bi-LSTM) + Rule-based (*intent keywords & Pos slots*)	*0.89*	*0.88*	0.88
Hybrid-3: RNN (Bi-LSTM) + Rule-based (*intent keywords & all slots*)	*0.90*	*0.90*	0.90
Separate-0: Seq2one LSTM	*0.87*	*0.86*	0.86
Separate-1: Seq2one Bi-LSTM	*0.88*	*0.88*	0.88
Separate-2: Seq2one Bi-LSTM + Attention	*0.88*	*0.88*	0.88
Separate-3: Seq2one Bi-LSTM + AttentionWithContext	*0.89*	*0.89*	0.89
Joint-1: Seq2seq Bi-LSTM (*uttr-level intents & slots*)	*0.88*	*0.87*	0.87
Joint-2: Seq2seq Bi-LSTM (*uttr-level intents & slots & intent keywords*)	*0.89*	*0.88*	0.88
Hierarchical & Separate-0 (LSTM)	*0.88*	*0.87*	0.87
Hierarchical & Separate-1 (Bi-LSTM)	*0.90*	*0.90*	0.90
Hierarchical & Separate-2 (Bi-LSTM + Attention)	*0.90*	*0.90*	0.90
Hierarchical & Separate-3 (Bi-LSTM + AttentionWithContext)	*0.90*	*0.90*	0.90
Hierarchical & Joint-2 (*uttr-level intents & slots & intent keywords*)	*0.91*	*0.90*	**0.91**

Table 4: Intent-wise Performance Results of Utterance-level Intent Detection

AMIE Scenario	**Intent Type**	**Our Intent Detection Models**						**Dialogflow Intent Detection**		
		Baseline (Hybrid-0)			**Best** (H-Joint-2)					
		Prec	*Rec*	**F1**	*Prec*	*Rec*	**F1**	*Prec*	*Rec*	**F1**
Finishing the Trip	Stop	*0.88*	*0.91*	0.90	*0.93*	*0.91*	0.92	*0.89*	*0.90*	0.90
	Park	*0.96*	*0.87*	0.91	*0.94*	*0.94*	0.94	*0.95*	*0.88*	0.91
	PullOver	*0.95*	*0.96*	0.95	*0.97*	*0.94*	0.96	*0.95*	*0.97*	0.96
	DropOff	*0.90*	*0.95*	0.92	*0.95*	*0.95*	0.95	*0.96*	*0.91*	0.93
Dest/Route	SetDest	*0.70*	*0.88*	0.78	*0.89*	*0.90*	0.89	*0.84*	*0.91*	0.87
	SetRoute	*0.80*	*0.71*	0.75	*0.86*	*0.89*	0.88	*0.83*	*0.86*	0.84
Speed	GoFaster	*0.86*	*0.89*	0.88	*0.89*	*0.90*	0.90	*0.94*	*0.92*	0.93
	GoSlower	*0.92*	*0.84*	0.88	*0.89*	*0.86*	0.88	*0.93*	*0.87*	0.90
Others	OpenDoor	*0.95*	*0.95*	0.95	*0.95*	*0.95*	0.95	*0.94*	*0.93*	0.93
	Other	*0.92*	*0.72*	0.80	*0.83*	*0.81*	0.82	*0.88*	*0.73*	0.80
	Overall	*0.86*	*0.85*	**0.85**	*0.91*	*0.90*	**0.91**	*0.90*	*0.89*	**0.89**

3.2 Slot Filling and Intent Keyword Extraction Experiments

Slot filling and intent keyword extraction results are given in Table 5 and Table 6, respectively. For slot extraction, we reached 0.96 overall F1-score using seq2seq Bi-LSTM model, which is slightly better than using LSTM model. Although the overall performance is slightly improved with Bi-LSTM model, relatively problematic *Object*, *Time Guidance*, *Gesture/Gaze* slots F1-score performances increased from 0.80 to 0.89, 0.80 to 0.85, and 0.87 to 0.92, respectively. Note that with Dialogflow, we reached 0.92 overall F1-score for the entity/slot filling task on the same dataset. As you can see, our models reached significantly higher F1-scores than the Dialogflow for 6 out of 7 slot types (except *Time Guidance*).

Table 5: Slot Filling Results (10-fold CV)

	Our Slot Filling Models						**Dialogflow Slot Filling**		
	Se2qseq LSTM			**Se2qseq Bi-LSTM**					
Slot Type	*Prec*	*Rec*	**F1**	*Prec*	*Rec*	**F1**	*Prec*	*Rec*	**F1**
Location	*0.94*	*0.92*	0.93	*0.96*	*0.94*	0.95	*0.94*	*0.81*	0.87
Position/Direction	*0.92*	*0.93*	0.93	*0.95*	*0.95*	0.95	*0.91*	*0.92*	0.91
Person	*0.97*	*0.96*	0.97	*0.98*	*0.97*	0.97	*0.96*	*0.76*	0.85
Object	*0.82*	*0.79*	0.80	*0.93*	*0.85*	0.89	*0.96*	*0.70*	0.81
Time Guidance	*0.88*	*0.73*	0.80	*0.90*	*0.80*	0.85	*0.93*	*0.82*	0.87
Gesture/Gaze	*0.86*	*0.88*	0.87	*0.92*	*0.92*	0.92	*0.86*	*0.65*	0.74
None	*0.97*	*0.98*	0.97	*0.97*	*0.98*	0.98	*0.92*	*0.98*	0.95
Overall	*0.95*	*0.95*	**0.95**	*0.96*	*0.96*	**0.96**	*0.92*	*0.92*	**0.92**

Table 6: Intent Keyword Extraction Results (10-fold CV)

Keyword Type	***Prec***	***Rec***	**F1**
Intent	*0.95*	*0.93*	0.94
Non-Intent	*0.98*	*0.99*	0.99
Overall	*0.98*	*0.98*	**0.98**

3.3 Speech-to-Text Experiments for AMIE: Training and Testing Models on ASR Outputs

For transcriptions, utterance-level audio clips were extracted from the passenger-facing video stream, which was the single source used for human transcriptions of all utterances from passengers, AMIE agent and the game master. Since our transcriptions-based intent/slot models assumed perfect (at least close to human-level) ASR in the previous sections, we experimented with more realistic scenario of using ASR outputs for intent/slot modeling. We employed Cloud Speech-to-Text API to obtain ASR outputs on audio clips with passenger utterances, which were segmented using transcription time-stamps. We observed an overall word error rate (WER) of 13.6% in ASR outputs for all 20 sessions of AMIE.

Considering that a generic ASR is used with no domain-specific acoustic models for this moving vehicle environment with in-cabin noise, the initial results were quite promising for us to move on with the model training on ASR outputs. For initial explorations, we created a new dataset having utterances with commands using ASR outputs of the in-cabin data (20 sessions with 1331 spoken utterances). Human transcriptions version of this set is also created. Although the dataset size is limited, both slot/intent keyword extraction models and utterance-level intent recognition models are not severely affected when trained and tested (10-fold CV) on ASR outputs instead of manual transcriptions. See Table 7 for the overall F1-scores of the compared models.

Table 7: F1-scores of Models Trained/Tested on Transcriptions vs. ASR Outputs

	Train/Test on Transcriptions			**Train/Test on ASR Outputs**		
Slot Filling & Intent Keywords	**ALL**	*Singleton*	*Dyad*	**ALL**	*Singleton*	*Dyad*
Slot Filling	0.97	*0.96*	*0.96*	0.95	*0.94*	*0.93*
Intent Keyword Extraction	0.98	*0.98*	*0.97*	0.97	*0.96*	*0.96*
Slot Filling & Intent Keyword Extraction	**0.95**	*0.95*	*0.94*	**0.94**	*0.92*	*0.91*
Utterance-level Intent Detection	**ALL**	*Singleton*	*Dyad*	**ALL**	*Singleton*	*Dyad*
Hierarchical & Separate	0.87	*0.85*	*0.86*	0.85	*0.84*	*0.83*
Hierarchical & Separate + Attention	0.89	*0.86*	*0.87*	0.86	*0.84*	*0.84*
Hierarchical & Joint	**0.89**	*0.87*	*0.88*	**0.87**	*0.85*	*0.85*

Singleton versus Dyad Sessions. After the ASR pipeline described above is completed for all 20 sessions of AMIE in-cabin dataset (*ALL* with 1331 utterances), we repeated all our experiments with the subsets for 10 sessions having single passenger (*Singletons* with 600 utterances) and remaining 10 sessions having two passengers (*Dyads* with 731 utterances). We observed overall WER of 13.5% and 13.7% for *Singletons* and *Dyads*, respectively. The overlapping speech cases with slightly more conversations going on (longer transcriptions) in *Dyad* sessions compared to the *Singleton* sessions may affect the ASR performance, which may also affect the intent/slots models performances. As shown in Table 7, although we have more samples with *Dyads*, the performance drops between the models trained on transcriptions vs. ASR outputs are slightly higher for the *Dyads* compared to the *Singletons*, as expected.

4 Discussion and Conclusion

We introduced AMIE, the intelligent in-cabin car agent responsible for handling certain AV-passenger interactions. We develop hierarchical and joint models to extract various passenger intents along with relevant slots for actions to be performed in AV, achieving F1-scores of 0.91 for intent recognition and 0.96 for slot extraction. We show that even using the generic ASR with noisy outputs, our models are still capable of achieving comparable results with models trained on human transcriptions. We believe that the ASR can be improved by collecting more in-domain data to obtain domain-specific acoustic models. These initial models will allow us to collect more speech data via bootstrapping with the intent-based dialogue application we have built, and the hierarchy we defined will allow us to eliminate costly annotation efforts in the future, especially for the word-level slots and intent keywords. Once enough domain-specific multi-modal data is collected, our future work is to explore training end-to-end dialogue agents for our in-cabin use-cases. We are planning to exploit other modalities for improved understanding of the in-cabin dialogue as well.

Acknowledgments. We would like to show our gratitude to our colleagues from Intel Labs, especially Cagri Tanriover for his tremendous efforts in coordinating and implementing the vehicle instrumentation to enhance multi-modal data collection setup (as he illustrated in Fig. 1), John Sherry and Richard Beckwith for their insight and expertise that greatly assisted the gathering of this UX grounded and ecologically valid dataset (via scavenger hunt protocol and WoZ research design). The authors are also immensely grateful to the members of GlobalMe, Inc., particularly Rick Lin and Sophie Salonga, for their extensive efforts in organizing and executing the data collection, transcription, and certain annotation tasks for this research in collaboration with our team at Intel Labs.

References

1. Chung, J., Gülçehre, Ç., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. CoRR **abs/1412.3555** (2014), http://arxiv.org/abs/1412.3555
2. Crowston, K.: Amazon mechanical turk: A research tool for organizations and information systems scholars. In: Bhattacherjee, A., Fitzgerald, B. (eds.) Shaping the Future of ICT Research. Methods and Approaches. pp. 210–221. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
3. Eric, M., Krishnan, L., Charette, F., Manning, C.D.: Key-value retrieval networks for task-oriented dialogue. In: Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue. pp. 37–49. Association for Computational Linguistics (2017). https://doi.org/10.18653/v1/W17-5506, http://aclweb.org/anthology/W17-5506
4. Fukui, M., Watanabe, T., Kanazawa, M.: Sound source separation for plural passenger speech recognition in smart mobility system. IEEE Transactions on Consumer Electronics **64**(3), 399–405 (Aug 2018). https://doi.org/10.1109/TCE.2018.2867801
5. Hakkani-Tur, D., Tur, G., Celikyilmaz, A., Chen, Y.N.V., Gao, J., Deng, L., Wang, Y.Y.: Multi-domain joint semantic frame parsing using bi-directional rnn-lstm. ISCA (June 2016), https://www.microsoft.com/en-us/research/publication/multijoint/
6. Hansen, J.H., Angkititrakul, P., Plucienkowski, J., Gallant, S., Yapanel, U., Pellom, B., Ward, W., Cole, R.: Cu-move: Analysis & corpus development for interactive in-vehicle speech systems. In: Seventh European Conference on Speech Communication and Technology (2001)
7. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
8. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. pp. 427–431. Association for Computational Linguistics (April 2017)
9. Liu, B., Lane, I.: Joint online spoken language understanding and language modeling with recurrent neural networks. CoRR **abs/1609.01462** (2016), http://arxiv.org/abs/1609.01462
10. Meng, Z., Mou, L., Jin, Z.: Hierarchical rnn with static sentence-level attention for text-based speaker change detection. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 2203–2206. CIKM '17, ACM, New York, NY, USA (2017). https://doi.org/10.1145/3132847.3133110, http://doi.acm.org/10.1145/3132847.3133110
11. Mesnil, G., Dauphin, Y., Yao, K., Bengio, Y., Deng, L., Hakkani-Tur, D., He, X., Heck, L., Tur, G., Yu, D., Zweig, G.: Using recurrent neural networks for slot filling in spoken language understanding. Trans. Audio, Speech and Lang. Proc. **23**(3), 530–539 (Mar 2015). https://doi.org/10.1109/TASLP.2014.2383614, http://dx.doi.org/10.1109/TASLP.2014.2383614
12. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. CoRR **abs/1301.3781** (2013)
13. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings

of the 26th International Conference on Neural Information Processing Systems - Volume 2. pp. 3111–3119. NIPS'13, Curran Associates Inc., USA (2013), http://dl.acm.org/citation.cfm?id=2999792.2999959
14. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP). pp. 1532–1543 (2014), http://www.aclweb.org/anthology/D14-1162
15. Raffel, C., Ellis, D.P.W.: Feed-forward networks with attention can solve some long-term memory problems. CoRR **abs/1512.08756** (2015), http://arxiv.org/abs/1512.08756
16. Ravuri, S., Stolcke, A.: Recurrent neural network and lstm models for lexical utterance classification. In: Proc. Interspeech. pp. 135–139. ISCA - International Speech Communication Association, Dresden (September 2015)
17. Schuster, M., Paliwal, K.: Bidirectional recurrent neural networks. Trans. Sig. Proc. **45**(11), 2673–2681 (Nov 1997). https://doi.org/10.1109/78.650093, http://dx.doi.org/10.1109/78.650093
18. Sherry, J., Beckwith, R., Arslan Esme, A., Tanriover, C.: Getting things done in an autonomous vehicle. In: Social Robots in the Wild Workshop at the 13th Annual ACM/IEEE International Conference on Human-Robot Interaction (HRI) (March 2018), http://socialrobotsinthewild.org/wp-content/uploads/2018/02/HRI-SRW_2018_paper_3.pdf
19. Stolcke, A., Droppo, J.: Comparing human and machine errors in conversational speech transcription. CoRR **abs/1708.08615** (2017), http://arxiv.org/abs/1708.08615
20. Wang, P., Sibi, S., Mok, B., Ju, W.: Marionette: Enabling on-road wizard-of-oz autonomous driving studies. In: Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction. pp. 234–243. HRI '17, ACM, New York, NY, USA (2017). https://doi.org/10.1145/2909824.3020256
21. Wen, L., Wang, X., Dong, Z., Chen, H.: Jointly modeling intent identification and slot filling with contextual and hierarchical information. In: Huang, X., Jiang, J., Zhao, D., Feng, Y., Hong, Y. (eds.) Natural Language Processing and Chinese Computing. pp. 3–15. Springer International Publishing, Cham (2018)
22. Xiong, W., Droppo, J., Huang, X., Seide, F., Seltzer, M., Stolcke, A., Yu, D., Zweig, G.: Achieving human parity in conversational speech recognition. CoRR **abs/1610.05256** (2016), http://arxiv.org/abs/1610.05256
23. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.: Hierarchical attention networks for document classification. In: Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 1480–1489 (2016). https://doi.org/10.18653/v1/N16-1174, http://www.aclweb.org/anthology/N16-1174
24. Zhang, X., Wang, H.: A joint model of intent determination and slot filling for spoken language understanding. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence. pp. 2993–2999. IJCAI'16, AAAI Press (2016), http://dl.acm.org/citation.cfm?id=3060832.3061040
25. Zheng, Y., Liu, Y., Hansen, J.H.L.: Navigation-orientated natural spoken language understanding for intelligent vehicle dialogue. In: 2017 IEEE Intelligent Vehicles Symposium (IV). pp. 559–564 (June 2017). https://doi.org/10.1109/IVS.2017.7995777
26. Zhou, Q., Wen, L., Wang, X., Ma, L., Wang, Y.: A hierarchical lstm model for joint tasks. In: Sun, M., Huang, X., Lin, H., Liu, Z., Liu, Y. (eds.) Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data. pp. 324–335. Springer International Publishing, Cham (2016)