Knowledge-Based Systems 208 (2020) 106428

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Multi-turn intent determination and slot filling with neural networks and regular expressions

Waheed Ahmed Abro [a,*], Guilin Qi [a], Zafar Ali [a], Yansong Feng [b], Muhammad Aamir [c]

[a] School of Computer Science and Engineering, Southeast University, Nanjing, China
[b] Institute of Computer Science and Technology, Peking University, Beijing, China
[c] Department of Computer, Huanggang Normal University, Huanggang, China

ARTICLE INFO

Article history:
Received 29 January 2020
Received in revised form 11 September 2020
Accepted 15 September 2020
Available online 18 September 2020

Keywords:
Natural language understanding
Intent determination
Slot filling
Regular expressions
Human–computer interaction

ABSTRACT

Intent determination and slot filling are two prominent research areas related to natural language understanding (NLU). In a multi-turn NLU system, contextual information from dialogue history is exploited to mitigate the ambiguity of user utterance. State-of-the-art models employ memory networks to encode dialogue context, which is used by neural networks for determining user intent and associated slots. However, these methods rely on a large amount of labelled data, whereas we often have limited labelled data. To address this problem, we propose a multi-task learning model based on neural networks and regular expressions (REs), to jointly perform intent determination and slot filling tasks. The proposed model integrates neural networks with REs to encode domain knowledge and handle cases with a limited amount of labelled data in an end-to-end trainable manner. More specifically, the model employs a pre-trained BERT model to obtain contextual word representations of user utterances. These representations are utilized by a memory network to encode multi-turn information which is shared by the tasks. Furthermore, the convolutional neural network (CNN) and the recurrent neural network (RNN) are applied to contextual word representations and dialogue context for intent determination and slot filling tasks, respectively. These neural networks are then combined with REs which encode domain knowledge about a particular intent or slot value. Finally, the two neural networks are trained simultaneously by minimizing the joint loss. Extensive experiments on Key-Value Retrieval and Frames datasets show that the proposed model outperforms baseline methods in both tasks while requiring modest human effort.

1. Introduction

Goal-oriented dialogue systems help users to solve a given task using natural language, such as finding a particular location, booking a ticket, or sending a message. Natural Language Understanding (NLU) module is a critical component of such systems, which converts the user utterance into a task-specific semantic representation. The main tasks of NLU are intent determination and slot filling [1]. Intent determination predicts the user intent, and slot filling fills the set of arguments or slots corresponding to a semantic frame. For instance, *"Please book a trip to New York from Mannheim"* as depicted in Fig. 1, with In/Out/Begin (IOB) representation. In this example, the intent of the user is to book a trip, *"New York"* and *"Mannheim"* fills the associated destination and origin city slot, respectively. Other words not corresponding to slots are tagged with the null label "O". The results from intent determination and slot filling are then used by the dialogue manager to form a query for the back-end and respond to the user accordingly.

Some prior studies [2–6] have focused on single-turn NLU in which the system receives only one utterance at a time and predicts user intent and slot labels. However, goal-oriented dialogue systems require both user and system to take multiple turns of back-and-forth interactions to accomplish a specific user goal. In multi-turn NLU, the user, as well as the system can refer to the entities of previous dialogue turn, resulting in contextual ambiguity [7]. For example, "all three" in an utterance can represent a number of activities, days, or tickets, generating ambiguity in understanding the semantics behind.

To overcome this problem, incorporating prior dialogue history has shown effective results in resolving such ambiguities of the utterances [8–10]. For instance, Chen et al. [8] employed a memory network to incorporate prior dialogue turns. In contrast, research studies [9,10] exploited the temporal order of the utterances to encode dialogue history in chronological order. The aforementioned methods generate promising results when

* Corresponding author.
E-mail addresses: waheed@seu.edu.cn (W.A. Abro), gqi@seu.edu.cn (G. Qi), fengyansong@pku.edu.cn (Y. Feng), aamirshaikh86@hotmail.com (M. Aamir).

https://doi.org/10.1016/j.knosys.2020.106428
0950-7051/

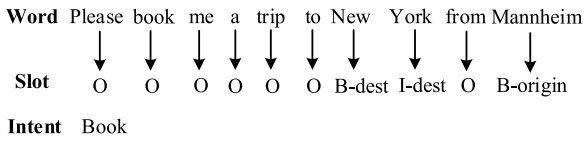

Fig. 1. An example of utterance with IOB representation for intent and slot values annotation.

there is a large amount of labelled data. Unfortunately, such labelled data is not always available. On the other hand, large-scale pre-trained language models, such as BERT (Bidirectional Encoder Representations from Transformers) [11] and XLNet [12] have achieved the state of the art performance on NLU tasks after simple fine-tuning. For instance, BERT is pre-trained with the transformer encoder on the large-scale unlabelled text and encodes the context of words from both forward and backward directions. As a result, the pre-trained BERT model can be then fine-tuned with comparatively small and task-specific training data on downstream tasks such as text classification [13]. Therefore, these large scale pre-trained language models have supported the improvement of language understanding tasks, even in the absence of sizeable datasets. Furthermore, regular expressions that are based on the hand-crafted rules, do not require training data to generate output [14].

Regular expressions (REs) are widely used for information extraction, named entity recognition, sentence classification, and slot tagging tasks [14,15]. REs consists of human-defined rules which are concise, tunable, and do not require much training data to generate. Due to the vast diversity in the user utterances, it is impossible to create a RE to identify particular slots and sentences with perfect accuracy. Therefore, instead of abandoning REs in favour of neural network approaches for sentence classification and slot tagging tasks, we can combine REs with neural networks. REs can complement the neural networks and can regulate the output of neural networks, especially in the absence of enough training data. More importantly, REs provide a mechanism for domain experts to encode domain knowledge and guide neural network models to capture desired patterns. REs can be applied at sentence-level and token-level. At the sentence-level, REs can be applied to search for key phrases in user utterance for particular user intent. At the token-level, REs can be employed to detect word sequence in user utterance and assign word-level RE label.

In light of the above intuition, this paper presents a Neural Network with Regular Expression (NN-RE) model, which exploits the expressive power of regular expressions along with the robustness of neural networks to predict user intents and slots labels. Furthermore, the correlation between intent determination and slot filling tasks makes them suitable for multi-task learning. The proposed model obtains contextual word representations of current utterances from the pre-trained BERT model. These representations along with dialogue context are utilized by dilated CNN to capture local features with a wider context for intent determination. At the same time, the model employs RNN to model the temporal relationship between word representations for slot filling task. More specifically, the model first employs a memory network to encode a dialogue context. The dialogue context vector is then shared by CNN and RNN for optimizing the objectives of slot filling and intent determination simultaneously via joint learning. Finally, regular expressions are utilized to complement CNN and RNN by encoding domain knowledge and handling cases with a limited amount of labelled data.

The main contributions of this study are summarized as follows:

- We propose a model that exploits the domain knowledge encoded in regular expressions to improve the performance of neural networks, especially in limited training data setting.
- We present a multi-task architecture that extracts the contextual word features from the pre-trained language model and employs CNN and RNN structures for modelling intent determination and slot filling tasks, simultaneously.
- We conducted extensive experiments on two real-world datasets, and evaluated the results of the proposed model against baseline competitors in terms of precision, recall, and F1 score metrics.

This work is a significant extension of our previous research [16] in the following ways. First, we adapt the model for slot filling task; second, we propose a multi-task architecture for intent determination and slot filling employing CNN and RNN to capture local and temporal features; third, we create a set of regular expressions for each task; and finally, we conduct extended experiments on two datasets and evaluate the results of the proposed model.

The rest of this paper is organized as follows: Section 2 presents related work. The overall architecture of the model is described in Section 3. Section 4 discusses the datasets, creation of regular expressions, training setting, baseline methods, and evaluation metrics. Section 5 presents results analysis and discussion followed by the conclusions of our research work in Section 6.

2. Related work

In this section, we provide an overview of the state of the art research on intent determination and slot filling tasks. Specifically, we analyse how existing methods have jointly modelled these tasks, incorporated human knowledge, utilized pre-trained language models, and contextual information for NLU tasks.

2.1. Natural language understanding

Deep learning models such as CNNs and RNNs have been extensively adopted in a wide range of NLU tasks, including intent determination [17–21] and slot filling [2,4,22]. For the intent determination task, deep belief networks (DBNs) [17] have been used to generate features from unlabelled data. These features are then used by SVM for performing the classification task. Deep belief networks have shown powerful feature extraction capabilities for classification tasks but training DBNs have proven to be computationally expensive because it is hard to parallelize DBNs training across machines for more massive datasets. To tackle the scalability issue of DBNs, Deng and Yu have proposed deep convex networks (DCNs) [23]. In this connection, Tur et al. [18] have used (DCNs) for semantic utterance classification and achieved higher accuracy. On the contrary, Zhang et al. [19] have employed CNN architecture as a feature extractor for capturing the semantics of user speech. Recently, RNNs architectures have shown excellent performance on the intent determination task, they take each word of a sentence and use a series of hidden units to temporally model an entire sentence [20,21]. Furthermore, Lin and Xu [24] proposed SofterMax and deep novelty detection (SMDN), a post-processing method for detecting unknown intent via pre-trained deep neural network classifiers. On the other hand, Howard and Cambria [25] discussed the importance of the intention-awareness in the human-centric environment where the intentions of actors' are integrated into the surrounding environment. Intention-aware (IA) based systems lessen the informational burden on humans without degrading effectiveness. In this connection, Liu and Zhang [26] have proposed a

method called WebIA, which infers human intentions through web queries. The WebIA method establishes an active knowledge database for robots to perform context-specific intention awareness.

The slot filling task is different from an intent determination as it generates output sequence, hence RNN performs well in such scenarios [2,22]. For example, Yao et al. [22] proposed a Recurrent Neural Network Language Models (RNN-LMs) architecture, which takes words as input and predicts slot labels rather than words on the output side. Similarly, Mesnil et al. [2] used a single RNN model for generating semantic tags sequentially by reading each word one by one. However, this approach can generate semantic tags equal to the number of words in the utterance. This limitation is overcome by using encoder–decoder architecture containing two separate RNN for encoding input and decoding output [4]. The advantage of encoder–decoder models is that it allows a system to match input words and output semantic tags with different lengths without considering alignment.

In recent years, various research works have jointly modelled intent determination and slot filling tasks. For example, Hakkani-Tur et al. [3] proposed a single RNN model for joint modelling of domain detection, intent determination, and slots filling. The RNN takes a sequence of words as input and generates semantic tags, and the last hidden states of RNN are used for intent and domain detection. Liu and Lane [4] further proposed the RNN encoder–decoder model for jointly modelling both tasks. First, hidden states information from slot filling RNN is consolidated, and then the attention model is used to generate its intent. Moreover, Goo et al. [5] proposed a slot-gated mechanism, which introduces an additional gate in LSTM that leverages the intent context vector for modelling slot-intent relationships to improve slot filling performance. On the other hand, Ramón López-Cózar [27] proposed a technique that considered knowledge of semantic frames and knowledge about the words in the application domain for improving the performance of the spoken dialogue systems.

Moreover, models encoding contextual information from dialogue history have shown higher generalization ability [8–10, 28]. For instance, Chen et al. [8] have used RNNs to encode user and system previous utterances and store them in memory. These memory embeddings are then matched with a current utterance vector using cosine similarity to obtain attention distribution over dialogue history. These attention distributions are then aggregated with memory embeddings to incorporate information from history. Bapna et al. [9] further extended the memory network by adding a session encoder to embed the context in chronological order. Encoding contextual information in sequential order results in a reduction of frame error rate. In the same direction, Su et al. [10] introduced a time-aware attention mechanism to pay more attention to recent utterances of dialogue by decaying attention weights over time. On the other hand, Kim et al. [28] employed two separate memories for the encoding system and user utterances differently depending on the speaker. Dual Memory Networks obtained significant performance improvement over the state-of-the-art contextual slot tagging models. Recently, Firdaus et al. [29] proposed a hierarchical convolutional neural network and hierarchical convolutional recurrent neural network for jointly modelling of intent detection and slot filling tasks.

Our model differs from the above-mentioned works as we propose a multi-task model employing dilated CNN for capturing local features and RNN for modelling temporal relationships for intent determination and slot filling tasks, respectively. Furthermore, the proposed model uses CNN for encoding a dialogue context instead of RNN, which is computationally expensive because the next output depends on the previous time step. The dialogue context representation is shared between the tasks. Moreover, the proposed model leverages REs to encode domain knowledge about a particular intent or slot value into the training of the model.

2.2. Pre-trained language models

The pre-trained language models such as ELMo [30], ULMFiT [13], GPT [31], BERT [11], XLNet [12], and more recently UNILM [32], MT-DNN [33], and ERNIE [34] trained on large unlabelled corpora are dominating in natural language understanding leaderboards such as GLUE [35] and SuperGLUE [36]. The ELMo [30] model learns deep contextualize word representation by training two-layer bidirectional LSTM with language modelling objective on the large text. The downstream model combines representations from all BiLSTM outputs using downstream task-specific weights. On the contrary, GPT [31] model adopted a discriminative fine-tuning approach when it is applied on downstream tasks. For every task, the task-specific layer is added on top of the pre-trained model and the whole model is fine-tuned with labelled data. In the same direction, ULMFiT [13] model employed discriminative fine-tuning with triangular learning rates. The ULMFiT uses gradual unfreezing of pre-trained parameters to retain the previous information and obtains state-of-the-art results for text classification tasks. An extensive study of the pre-trained models for text classification was done by Minaee et al. [37]. The study provides a quantitative analysis of the performance of different deep learning models for text classification on popular benchmarks. On the other hand, Peters et al. [38] explored how to adapt pre-trained models for down streaming tasks. They focused on feature extraction and fine-tuning of pre-trained model adaptation methods and performs experiments on diverse NLP tasks. Peters et al. [38] concluded that fine-tuning and feature extraction approaches have comparable performance in most cases and their relative performance depends on the similarity between the pre-training and the target tasks. In this paper, we have adopted a feature extraction method for joint intent determination and slot filling task. The proposed model extracts contextual representations of the words from the pre-trained BERT model which are utilized by the RNN and CNN network for determining user intent and associated slots.

2.3. Combination of neural networks with knowledge

Putting constraint on the output of the deep neural networks, to comply with predefined rules has been an active research topic [39–41]. In this connection, Hu et al. [39] used first-order logic rules to adjust the output probability of a neural network and then train the student network from the rule-regularized teacher network. Alashkar et al. [40] trained a neural network from examples and knowledge base rules by minimizing a joint loss of examples-based and rules-based network. Similarly, Luo et al. [41], incorporated knowledge of regular expressions into the training of a neural network as an input feature, as an attention guide, and as a regularization of neural network output. In contrast, Rule-Guided Embedding (RUGE) [42] was proposed that inject soft logic rules into learned knowledge graphs (KGs) embeddings.

On the other hand, Zhou et al. [43] proposed the model that leverages commonsense knowledge using a knowledge graph in neural dialogue systems. The model retrieves knowledge graphs for each user utterance and then encodes the graphs with an attention mechanism to augment the semantic information and thus supports a better understanding of utterances. Similarly, Guan et al. [44] have proposed the neural-based model that utilizes commonsense knowledge by multi-source attention to generate the story ending for a given story context. Recently, Young et al. [45] augmented the Seq2Seq framework with audio features of the user message for neural conversation generation and outperformed the audio-free models. The latent intention

dialogue model (LIDM) [46] was proposed that employs a discrete latent variable to learn the complex distribution of dialogue intentions. The latent variables represent the dialogue intentions, based on which a dialogue agent generates appropriate responses. The LIDM model was trained using supervised using reinforcement learning manners. Likewise, Xu et al. [47] proposed hierarchical encoder–decoder which exploited a discrete latent variable to learn dialogue intentions. The model augments latent intention inferred from the user utterance and trained end-to-end manner under unsupervised, semi-supervised, or reinforcement learning.

Likewise, Semantic Referee [48] was introduced that interacts with neural network classifier to extract qualitative features of the errors and suggest corrections. In contrast, Locascio et al. [49] proposed the LSTM model to generate regular expressions from the natural language specifications of REs. On the other hand, Zhang et al. [14] employed regular expressions for generating weak labels of the entity mentions from unlabelled data and then trained a neural network to predict those RE-generated weak labels.

Previous methods have not considered the task of intent determination and slot filling of NLU. Furthermore, we do not leverage REs to generate weak labels and we do not generate REs from neural networks. Instead, we use human-generated REs to improve the prediction ability of CNN and RNN for intent determination and slot filling tasks.

3. Model architecture

The proposed model architecture is depicted in Fig. 2. The model is divided into two main modules, namely: memory network encoder (Section 3.1), and intent determination and slot-filling module (Section 3.2). The memory network encoder module generates a vector representation of the dialogue history, which is later fed into the input of intent determination and slot-filling tasks. The intent determination and slot-filling tasks module use dilated CNN and RNN to determine user intent and extract associated slots of given utterance simultaneously, and it employs regular expression to complement neural networks.

3.1. Memory network encoder

In this section, we discuss how to generate a dialogue context vector from the prior dialogue turns. The proposed model encodes previous user and system utterances into vector space, and stores these vectors in the memory. Similarly, a current utterance is also encoded into a vector space, which is then compared with memory vectors for encoding knowledge using cosine attention. Further details of the memory network encoder working process are given in the following subsections.

3.1.1. Contextual word representation from BERT

To extract contextual word representations of user utterances, our model employs the pre-trained BERT [11] model, which has 12 transformers [50] encoders, 768 hidden states, and 12 self-attention heads. The BERT model is pre-trained on large-scale unlabelled text with training objectives of masked language modelling and the next sentence prediction. More specifically, the utterance representation from the BERT model is computed as follows: The user utterance is passed through the tokenizer layer, which splits the utterance into a list of tokens using WordPiece [51], and then combines three embeddings i.e., token, position, and segment to obtain a fixed-length vector. Furthermore, the special classification `[CLS]` token is inserted at the start of each utterance. Likewise, at the end of each utterance, a special `[SEP]` token is appended as a final token. The BERT encoder then computes the user utterance representation, i.e., hidden states for each token x_t as shown in Eq. (1):

$$H_t^n = BERT(x_{[CLS]}, x_1, \ldots, x_t, x_{[SEP]}) \tag{1}$$

where $H_t^n = (h_{[CLS]}^n, h_1^n, \ldots, h_t^n, h_{[SEP]}^n)$, n denotes number of BERT encoder layer, and h_t is the contextual representations of token t. For every word in the utterance, we obtain the word representations by summing the hidden states of last 4 layers $H_t = H_t^9 + H_t^{10} + H_t^{11} + H_t^{12}$ as suggested by Devlin et al. [11]. The contextual representations of each token are passed to CNN and LSTM layers to predict the corresponding user intent and slot labels. It is noteworthy that WordPiece tokenizer split tokens into sub tokens. For instance, 'quickest' split into two tokens 'quick', and '##est'. To tackle this problem in the slot filling task, we use the first sub-token representation of a word as input to the RNN model.

3.1.2. Utterance encoder

The utterance encoder uses a convolutional neural network (CNN) to encode current utterance c consists of n words: $c = w_1, w_2, \ldots, w_n$, into continuous space with dimension d. The current utterance is passed through BERT embedding layer to extract fixed sized features. Furthermore, we apply a 1-dimensional convolution on top of these features. Next, max pooling operation on each filter is performed to obtain a fixed length output d. The representation of current utterance u can be defined as:

$$\begin{aligned} c &= BERT(c), \\ u &= CNN_{inp}(c), \end{aligned} \tag{2}$$

where CNN_{inp} represents 1-dimensional convolution followed by max pooling operation for input utterance.

The 1-dimensional convolution operation convolves a filter W over the window of m words in the input sequence to produce a feature map. For instance, we generate feature p_i by applying filter W to the window of words $[x_i, \ldots, x_{i:i+m-1}]$ as:

$$\begin{aligned} p_i &= f(W \cdot x_{i:i+m-1} + b), \\ x_{i:i+m-1} &= x_i \oplus x_{i+1} \oplus \cdots \oplus x_{i+m-1}, \end{aligned} \tag{3}$$

where b is a bias, $\oplus$ is the vector concatenation operator, and f represents non-linear activation function. We apply this filter to all locations of the current utterance c, producing feature maps: $p = [p_1, p_2, \cdots, p_{n-m+1}]$. The model then applies max-pooling operation to extract the maximum value from each feature map into a single fixed d-dimension vector: $u = max(p)$.

3.1.3. Memory encoder

To store and incorporate dialogue history, the model takes previous dialogue turns i.e., $x_1, x_2, \ldots, x_i$, passes them through the BERT layer to obtain the word-level features for each turn. These features are given as input to CNN for generating memory vectors $m_1, m_2, \ldots, m_i$. Each memory vector possess same dimension d as the current utterance, as shown in Eq. (4).

$$\begin{aligned} x_i &= BERT(x_i), \\ m_i &= CNN_{mem}(x_i). \end{aligned} \tag{4}$$

The primary aim is to store previous turns knowledge into memory for future usage. To retrieve information from the memory, attention mechanism is applied to selectively pay attention to a specific history utterance. To do so, the model computes attention distribution a_i over the memory vectors $m_1, m_2, \ldots, m_i$ by taking the cosine similarity between each memory vector m_i and current utterance representation u followed by the softmax activation as shown in Eq. (5).

$$a_i = softmax(u^T m_i), \tag{5}$$

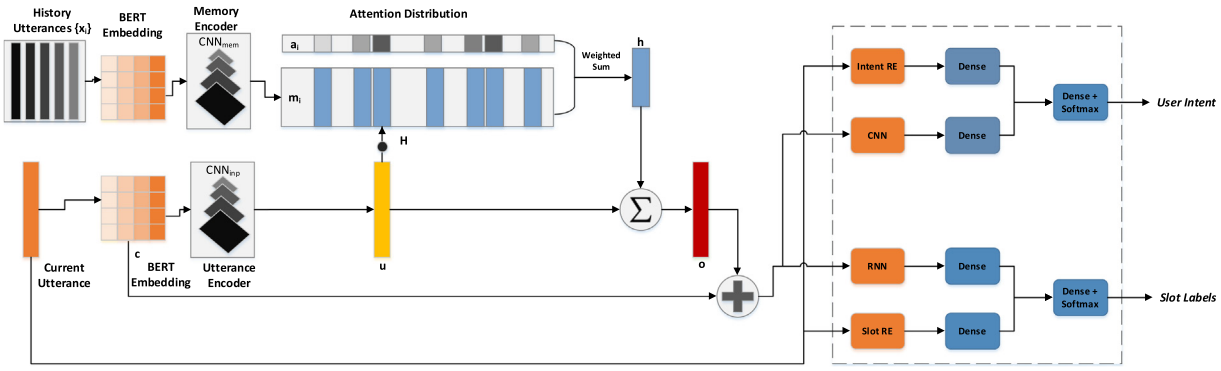

Fig. 2. Architecture of the proposed model for multi-turn intent determination and slot filling. A dialogue context vector and current utterance is shared by RNN and CNN network. REs are integrated with RNN and CNN to predict user intent and slot labels.

where $softmax(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}}$.

To accumulate knowledge from history, the history vector h is obtained by taking the weighted sum over the memory vectors m_i by the attention distribution a_i as presented in Eq. (6).

$$h = \sum_i a_i m_i \tag{6}$$

Finally, the history vector h and current utterance u are added to generate dialogue context vector o,

$$o = (h + u). \tag{7}$$

The dialogue context vector o is then passed to the intent determination and slot-filling module discussed in the following section.

3.2. Intent determination and slot filling module

For both intent determination and slot-filling tasks, the model leverages the dialogue context. More specifically, the model concatenates dialogue context vector o with user utterance representation obtained from the BERT embedding layer c; this input is then shared by both tasks. The intent determination is usually framed as a sentence classification problem; therefore, we use a convolutional neural network. Convolutional neural networks are designed to produce local and position-invariant features, and they capture local key-phrases in user sentences; hence perform well on sentence classification tasks [52,53]. On the other hand, the slot filling task is treated as a sequence labelling problem; therefore, we use a recurrent neural network (RNN). Recurrent neural networks can model temporal relationships, hence suitable for modelling sequential information [2,4]. Furthermore, we employ regular expressions to encode domain knowledge and handle cases with a limited amount of training data.

3.2.1. Slot filling via recurrent neural network

Recurrent neural networks (RNNs) are powerful architectures capable of capturing long-range dependencies via time-connection feedback [54]. Therefore, the proposed model uses a recurrent neural network (RNN) to model long-range dependencies. To overcome the vanishing gradient problem of RNNs, long short-term memory (LSTM) was designed [55]. LSTMs are the same as RNNs except that the hidden layer updates are replaced by memory cells. Fig. 4, illustrates the basic structure of the LSTM memory cell used in the model. Formally, the memory cell update process in the LSTM network is defined as follows:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\ \widetilde{c_t} &= tanh(W_c x_t + U_c h_{t-1} + b_c) \\ c_t &= f_t \odot c_{t-1} + i_t \odot \widetilde{c_t} \\ h_t &= o_t \odot tanh(c_t), \end{aligned} \tag{8}$$

where σ and $\odot$ denote sigmoid function and element-wise multiplication respectively. At time step t, there is an input gate i_t, a forget gate f_t, an output gate o_t, a cell state c_t, and a hidden state h_t. W, U, b are the parameters of LSTM. We place LSTM in forward and backward directions and concatenate two LSTMs outputs to access both past and future information for a given time. Input to LSTM is the dialogue context vector obtained from Eq. (7) and contextual word representation of current utterance as defined in Eq. (9).

$$\begin{aligned} \overrightarrow{h_t} &= \overrightarrow{LSTM}(o, c, \overrightarrow{h}_{t-1}) \\ \overleftarrow{h_t} &= \overleftarrow{LSTM}(o, c, \overleftarrow{h}_{t+1}) \\ s_r &= [\overrightarrow{h_t}, \overleftarrow{h_t}]. \end{aligned} \tag{9}$$

Where $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ are the hidden states of forward and backward passes in BLSTM, while c represents current utterance and o demonstrates dialogue context vector. To obtain a per-class score for slot filling, the representation s_r is passed through a fully connected network.

$$logit_{slot} = W_r(s_r), \tag{10}$$

where W_r is the weight matrix for a fully-connected network and $logit_{slot}$ is the unnormalized predictions of a model.

3.2.2. Regular expressions for slot filling

Regular Expressions (REs) are an algebraic notation that is used to describe search patterns we want to match. More specifically, REs search through the user sentences and return all texts which meet its search pattern. Existing research studies have exploited the significance of REs in different natural language processing tasks namely information extraction [56], sequence tagging [15] and named entity recognition [14] tasks. REs are often used to extract patterns from text based on pre-defined rules. However, REs can be combined with neural models instead of being just simple pattern matching. REs complement the robustness of neural networks by providing control of a rule-based system, especially in the absence of enough training data. Furthermore, REs facilitate domain experts to encode domain knowledge about particular user intent or particular structure of slot value. The

Utterance	Slot RE:	Match	RE-Label assigned
Trip to New York from Mannheim.	/to dst_city from or_city/	to New York from Mannheim	O B-dst_city I-dst_city O B-or_city
We are 3 adults.	/([0-9]+) adults/	3 adults	B-n_adult I-n_adults

Fig. 3. Slot RE examples with word-level labels assigned to the matched phrase.

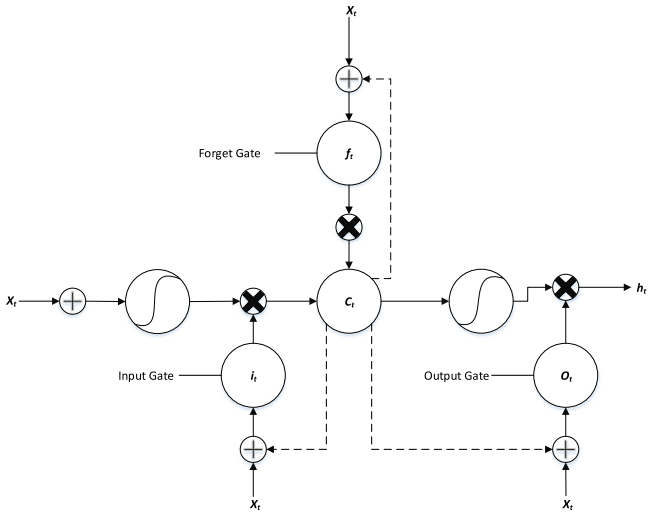

Fig. 4. Basic structure of LSTM memory cell.

proposed model incorporates knowledge expressed by REs into the training of CNN and RNN for the intent determination and slot filling.

For the slot filling task, we write regular expressions to match the specific structure of the slot being extracted. In Fig. 3, we illustrate some examples of slot REs. In first slot RE, "`or_city`" and "`dst_city`" represent word list of city names in the dataset. This RE matches "to New York from Mannheim" phrase. Slot RE module then assigns the RE label to each word of the matched phrase. The word-level labels are also converted into BIO format. For instance, `B-dst_city` and `I-dst_city` for New York destination city. In the second example, slot RE matches "3 adults" phrase of the utterance, "B-n_adults" and "I-n_adults" are the labels assigned to it.

To generate the slot REs score, a fully connected layer is applied to the REs search outcome. The output produced by slot REs is used to amend the output of RNN, which is fed as an input to a softmax layer for predicting IOB slot labels as defined in Eq. (11).

$$\hat{y}_{slot} = softmax(W_{slot}(logit_{slot} + W_{reg}R^k)). \tag{11}$$

Where $logit_{slot}$ is the logit obtained from Eq. (10) and W_{reg} represents a trainable weight matrix for REs while R^k is the REs search output containing 0-1 values for each slot, which indicates the absence or presence of at least one matched RE results in slot label k.

3.2.3. Intent determination using dilated CNNs

Traditional convolutional models incorporate local word context information into word representation, where the filter width specifies the local context size used. In order to increase effective context size, several convolutional layers are stacked in a hierarchical structure. However, this setting can incorporate context size linear to the entire depth of the network, therefore makes it harder for tasks that require longer history. To overcome such problems, we use dilated convolutions that can capture large context by skipping the number of nearby words in subsequent convolutional passes. In particular, a dilated convolution, also known as atrous convolution is a convolution operation in which the receptive field of the network grows exponentially with linear parameter accretion [6,57]. Besides, to enlarge the receptive field, dilation convolution inserts zeros between filter weights in the standard convolutional filter. For instance, the generic dilated convolution for dilations 1, 2 and 4 is shown in Fig. 6. The Dilated convolution operator of filter W_p and dilation factor d applied to m-gram $x_{i+m.d}$ with output p_i is defined as:

$$\begin{aligned} p_i &= f(W_p \cdot x_{i+m.d} + b), \\ x_{i+m.d} &= x_i \oplus x_{i+1.d} \oplus \cdots \oplus x_{i+m.d}, \end{aligned} \tag{12}$$

where $\oplus$ is the vector concatenation operator, d is the dilation factor, b is the bias, and f is the activation function. We use hyperbolic tangent (tanh) as the non-linear activation function. When the size of the dilation factor is 1, its equivalent to a regular convolution. Using larger dilation incorporate wider context into the representation of an input token p_i. We apply this filter with an exponentially increasing dilation $1, 2, 4$ and 8 to the concatenated vector of dialogue context o and current utterance representation c for extracting feature vectors. A max-pooling layer is then applied to feature vectors to induce final indicative feature vector s_c for intent determination as defined:

$$\begin{aligned} p &= [p_1, p_2, \cdots, p_{n-m.d}], \\ s_c &= max(p), \end{aligned} \tag{13}$$

The main idea is to apply 1-dimensional convolution with dilated filters for capturing important m-gram with the wider context. Then, apply max pooling over time to extract the most important m-grams for determining user intent. Next, the final state s_c of dilated CNN is passed through a fully connected network to obtain per-class scores for each intent.

$$logit_{in} = W_c(s_c), \tag{14}$$

where W_c is the weight matrix for the fully connected network.

3.2.4. Regular expressions for intent determination

For the intent determination task, we write regular expressions to search for key phrases in user utterance and assign RE intent label on the basis of key phrases. In Fig. 5, we demonstrate some examples of intent REs. In the first example, intent RE finds clue words i.e., "set reminder" in user utterance and assign *schedule* RE label. In the second example, intent RE matches "give me directions" keywords and assign *navigate* RE label.

To generate an intent REs score, a fully connected layer is applied to the REs assigned labels. The output produced by intent REs then is used to amend the output of dilated CNN which is utilized for determining the user intent as defined follows.

$$\hat{y}_{in} = softmax(W_{in}(logit_{in} + W_{reg}R^k)). \tag{15}$$

Where W_{in} represents a trainable weight matrix, $logit_{in}$ is the logit obtained from Eq. (14) and W_{reg} represents a trainable weight matrix for REs while R^k is the REs intent labels containing 0-1 values for each intent, which indicates the absence or presence of at least one matched RE results in intent label k.

Utterance	Intent RE:	Match	RE-Label assigned
set reminder for doctor's appointment	/set (a) ? reminder /	set reminder	schedule
give me directions to the nearest shopping center	/(give)(?:\w+){0,3}?(directions)/	give me directions	navigate

Fig. 5. Intent RE examples with the sentence-level label assigned to the matched utterance.

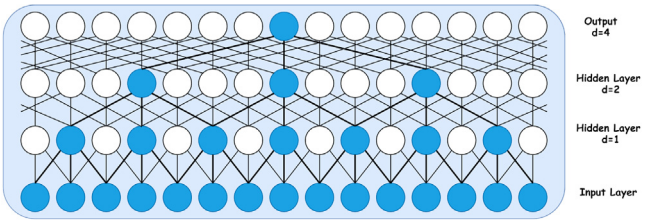

Fig. 6. A dilated convolution with dilation factor d = 1, 2, 4 and filter width = 3.

3.3. Joint optimization

For joint optimization, we use categorical cross-entropy loss for both intent determination ($\mathcal{L}_{in}$) and slot filling ($\mathcal{L}_{slot}$) tasks. The loss function for intent determination is $\mathcal{L}_{in}$, and for slot filling is $\mathcal{L}_{slot}$.

$$\begin{aligned} \mathcal{L}_{in} &= -\sum_{i=1}^{l} y_{in}^{i} log(\hat{y}_{in}^{i}) \\ \mathcal{L}_{slot} &= -\sum_{t}\sum_{n} y_{t}^{n} log(\hat{y}_{t}{}^{n}) \\ \mathcal{L} &= \mathcal{L}_{in} + \mathcal{L}_{slot}. \end{aligned} \tag{16}$$

Where l is the number of intent types, t is the number of words in the utterance, n is the number of slot labels, and y_{in}, y_t^n denote the ground truth of user intent and slot label for the tth word, respectively. The loss value minimized by the model is the sum of both individual losses. The overall optimization of the model is summarized in the Algorithm 1.

4. Experiments

4.1. The dataset

The ATIS [58] dataset is commonly used in literature for single turn intent determination and slot filling tasks. In this work, our problem is multi-turn NLU, thus, the ATIS dataset does not fit the experimental setup. Hence, we considered two publicly available datasets namely Frames [59] and Key-Value Retrieval [60] for multi-turn NLU problems. The statistics of these datasets are given in Table 2 and example dialogues are given in Table 1.

4.1.1. Key-value retrieval dataset

The Key-Value Retrieval (KVRET) dataset is composed of multi-turn dialogue between a driver and an In-car assistant.[1] A dialogue between the user and assistant starts with a user sentence that creates a dialogue domain using which the requirements of a user is achieved accordingly. This dataset contains three intents: calendar scheduling, point-of-interest navigation, and weather information. There are 15 slot types, each slot is stored with its value. Before using the dataset, we performed the following preprocessing steps: (1) annotating intent from session to sentence level by copying intent values. (2) obtaining word-level annotation by matching the slot values in the manual annotations with the corresponding sentence. We used IOB (Inside, Outside, Beginning) tag format, which is a common word-level annotation for slot filling tasks. Furthermore, we used 2425, 302, 304 dialogues for training, validation, and testing respectively.

Algorithm 1 Integrating RE into NN training

Input: Training data (x_t, U, C, R_0, R_1)
// x_t: set of history utterances
// U: current utterances surface form
// C: current utterances word representations
// R_0: expert defined regular expressions for slots
// R_1: expert defined regular expressions for intent
Initialize neural network parameter θ
foreach $((u,c) \in (U,C))$ **do**
  $o \leftarrow$ DialogueContext(x_t, c)
  $s_r \leftarrow$ RNN (o, c)
  $s_c \leftarrow$ DCNN (o, c)
  $logits_{slot} \leftarrow Dense(s_r)$
  $logits_{in} \leftarrow Dense(s_c)$
  foreach $(r_o \in R_0)$ **do**
    match $\leftarrow$ re.search(r_o, u)
    SlotLabel $\leftarrow$ match
  RESlot $\leftarrow$ Dense(SlotLabel)
  foreach $(r_1 \in R_1)$ **do**
    match $\leftarrow$ re.search(r_1, u)
    IntentLabel $\leftarrow$ match
  REIntent $\leftarrow$ Dense(IntentLabel)
  slots $\leftarrow$ softmax (Dense($logits_{slot}$, RESlot))
  intent $\leftarrow$ softmax(Dense($logits_{in}$, REIntent))
  Calculate loss using Eq. (16)
  Update parameter θ
Output: neural network model parameter

4.1.2. Frames dataset

Frames dataset[2] contains hotel and travel-booking dialogues collected in Wizard-of-Oz Scheme. It consists of 1369 multi-turn dialogues with an average of 15 turns per dialogue, for a total of 19986 turns. Each dialogue turn in the dataset is annotated with intent, slot types, and slot values. There are 20 intents and 28 slot types. Similar to the preprocessing steps adopted in the KVRET dataset, we obtained word-level tags by matching slot value with the concerned slot type in the sentences. Furthermore, the dataset contains a total of 11 participants dialogues, we selected two participants (ids 'U21E41CQP' and 'U231PNNA3') dialogues for testing and the other nine users dialogues are randomly divided into training (90%) and validation (10%) sets.

4.2. Creation of regular expressions

We used python `re` module for writing regular expressions. Our REs are written by a domain expert. It took less than 20 h to

[1] https://nlp.stanford.edu/blog/a-new-multi-turn-multi-domain-task-oriented-dialogue-dataset/

[2] https://datasets.maluuba.com/Frames

Table 1
Example dialogues from KVRET and Frames dataset.

Dataset	Speaker	Utterance	Slots	Intent
KVRET	Driver	What is the highest temperature for this week	weather_attribute, date	Weather
	Assistant	What location do you want to know?		
	Driver	Menlo Park, please	location	Weather
	Assistant	On Friday in Menlo Park it will be a high of 100f.		
	Driver	Get me directions to the nearest shopping mall	poi_type, distance	Navigate
Frames	User	Hi im from Leon and looking to get away	or_city	Inform
	Wizard	Do you have a special destination in mind		Request
	User	Maybe milan	dst_city	Inform
	Wizard	Any specific dates in mind		Request
	User	August 18th until September 2nd	str_date, end_date	Inform
	Wizard	I have a trip available departing on Aug 19 and returning on the 23rd	str_date, end_date	Offer

Table 2
Basic statistics of datasets.

	KVRET	Frames
Training dialogues	2425	1109
Validation dialogues	302	125
Test dialogues	304	135
Total turns	12732	19986
Avg. turns per dialogue	5.25	15
Intent type	3	20
Slot types	15	28

write all intent REs and slot REs for both datasets. We randomly selected 10 training dialogues to create REs, and the writing process of REs is completed when REs cover most of the cases with reasonable accuracy. After that, we used those REs throughout the experiments. Generally, more complex REs containing multiple REs groups and alternation lead to better precision but slightly lower coverage. Furthermore, it is much faster to apply many small REs containing few groups and alternation instead of one large RE containing a lot of groups and alternation [61]. Therefore, we constructed REs with few groups, and alternation details are given below.

For the intent determination task, we created 15 distinct intent REs containing on average 3 groups and 5 alternations for the KVRET dataset. For the Frames dataset, 25 intent REs containing 2.5 groups on average with 3 alternations are created. To write 40 intent REs, it took around 7 h. For the slot filling task, we created 30 slot REs with an average 2 groups and 6 alternations for the KVRET dataset and 35 slot REs with an average 2 groups and 4 alternations for the Frames dataset. It took us about 12.5 h to write 65 slot REs.

4.3. Training settings

The training setup of the proposed model consists of two kinds of settings namely: full training data setting and limited training data setting. In a full training data setting, we consider all training data to learn contextual features. Whereas, in limited training data setting only 20% of training data for each intent and slots are taken. Furthermore, for both settings, the proposed model is trained with Adam optimizer [62] using the default coefficients $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and fixed learning rate of 0.001. Besides, each update is computed through the batch size of 64 training samples. In addition, the number of epochs per batch is set to 150. Also, the model employed the pre-trained Bert-Base model[3] with 12 Transformer encoder layers, 768 hidden states, and 12 self-attention heads to obtain 768 dimension contextual word embeddings. Additionally, we use the memory size of 5 to store the knowledge from the previous five turns. Moreover, to avoid over-fitting in the model dropout is used as a regularizer. Furthermore, the rate of dropout is set to 0.5 for all dropout layers. Table 3 summarizes the details of hyper-parameters related to CNN and LSTM models used in this research study.

[3] https://storage.googleapis.com/bert_models/2020_02_20/uncased_L-12_H-768_A-12.zip

Table 3
Hyper-parameters of experiments.

Layer	Hyper-parameter	Size
BERT Embedding	Dimension	768
CNN	Number of filters	768
	Filter size	3
LSTM	hidden layer	100
DCNN	Number of filters	100
	Filter size	3
	Dilation rate	1, 2, 4, 8
Dropout	Dropout rate	0.5
	memory size	5
	Batch size	64
	Number of epochs	150
	Learning rate	0.001

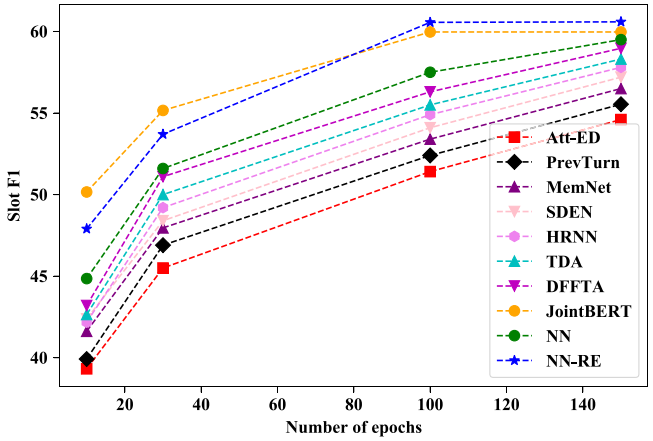

Fig. 7. Learning from limited training data results on Frames dataset for slot filling task.

4.4. Baseline methods

To assess the performance of the proposed model, we compared the results of the model against the following baseline methods.

1. Attention Encoder–Decoder: The model [4] encodes current utterance using bi-directional RNN-LSTM and generates the output using another uni-directional RNN-LSTM with attention. The number of units in the LSTM cell is set to

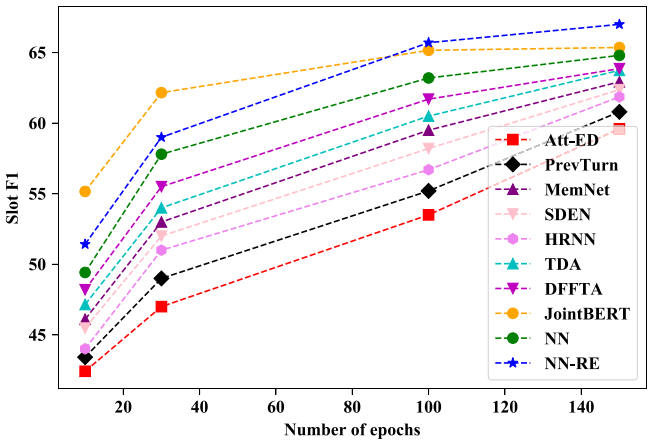

Fig. 8. Learning from limited training data results on KVRET dataset for slot filling task.

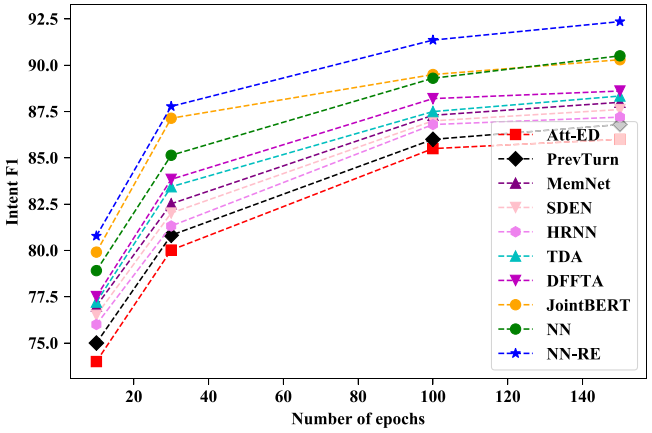

Fig. 9. Learning from limited training data results on KVRET dataset for intent determination task.

100 while the dropout rate of 0.2 is applied to avoid overfitting. In this model, we do not provide any contextual information. We used the online available code.[4]

2. Previous Turn Context: This architecture encodes previous turn and current utterance via bidirectional RNN-LSTM, which is given as an input to another RNN-LSTM for predicting the user intent and slots. The dimensions for the distributed word representations and the size of a hidden layer of LSTM are set to 100 each.
3. Memory Network (MemNet): The model architecture [8] encodes previous dialogue history and current utterances with attention and memory mechanisms. The dialogue context vector is fed as an additional input to the RNN-GRU tagger for predicting the user intent and slots. We trained this model with a memory size of 5 while the size of the hidden layer in LSTM is set to 100. We used the online available code[5] provided by the authors.
4. Sequential Dialogue Encoder Network (SDEN): This model [9] encodes the previous history into memory vectors using bidirectional RNN-GRU. The number of units in BiGRU is set to 128 (64 each direction), while the 128-dimensional context vector is produced by a feed-forward layer from the concatenated vector of memory vector and current utterance. These context vectors are combined using another bidirectional RNN-GRU to represent dialogue context encoding used by tagger architecture for intent determination and sequence tagging. We trained the SDEN model with a memory size of 5. We have utilized MemNet code[6] after making the necessary modifications.
5. Hierarchical Recurrent Neural networks (HRRN): The model architecture [7], encodes system dialogue acts instead of the system utterance for producing dialogue context. It employs hierarchical RNN for encoding dialogue context obtained by summarizing the content of the dialogue act and the user utterance vector. This representation is then used by bidirectional RNN-LSTM for slot tagging. The size of the hidden layer in LSTM is set to 100.
6. Time-Decay Attention (TDA): The Time Decay Attention model [10], encodes dialogue history into the memory using bidirectional RNN-LSTM. The model uses a time-decay attention mechanism to pay more attention to recent utterances of dialogue by decaying attention weights over time. This representation is used by RNN-LSTM as additional input for sequence tagging. We trained the model with a history length of 5. The size of the hidden layer in LSTM is set to 128. We utilized the online available code[7] provided by the authors.
7. Decay-Function-Free Time-Aware Attention (DFFTA): The model [63] takes the current utterance as input and fed into bidirectional RNN-LSTM to obtain the current utterance summary. To summarize the dialogue context, the model uses the time difference between history utterance and current utterance. The model automatically learns the time-decay function of the history by introducing a trainable distance vector to pay more attention to important history utterances based on distance with current utterance. The dialogue context summary is fed as an additional input into RNN-LSTM for sequence tagging. We trained the model with a history length of 5. The size of the hidden layer in LSTM is set to 128. We used the code[8] provided by the authors.
8. BERT for Joint Intent Classification and Slot Filling (Joint BERT): The model [64] fine-tunes the pre-trained BERT model to jointly perform intent classification and slot filling tasks. The model utilizes the hidden states of the first `[CLS]` token for predicting user intent. For slot filling, the model feeds the final hidden states of all other tokens to the softmax layer to classify the slot labels. To make results comparable with other models, the length of history is set to be 5. The dialogue history utterances are sum together to form a dialogue context vector. The dialogue context vector is concatenated with the last hidden states of the current utterance. The model is fine-tuned with Adam optimizer using a learning rate of $5e-5$ and a batch size of 128 as suggested by authors.

4.5. Evaluation metrics

To evaluate the results of the proposed model, We have utilized the most commonly used evaluation metrics such as precision, recall, and F1 score.

4 https://github.com/HadoopIt/rnn-nlu
5 https://github.com/yvchen/ContextualSLU
6 https://github.com/yvchen/ContextualSLU
7 https://github.com/MiuLab/Time-Decay-SLU
8 https://github.com/jgkimi/Decay-Function-Free-Time-Aware

Precision: Precision calculates the number of intent/slots predicted by the model as positive are actually positive, as given in the following equation.

$$Precision = \frac{\#true\_positives}{\#true\_positives + \#false\_positives} \quad (17)$$

Recall: Recall calculates the number of actual positive intent/slots the model capture through labelling as positive.

$$Recall = \frac{\#true\_positives}{\#true\_positives + \#false\_negatives} \quad (18)$$

F1 Score: F1 Score computes the harmonic mean of precision and recall, defined as follows:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (19)$$

5. Results and discussion

Our experiments aim to investigate, whether REs can improve the learning ability of neural networks when we have limited training data. To answer this question, we have performed experiments on a limited training data setting (Section 5.1), which considers only 20% of the training data. We also investigated, whether REs are still helpful when we have enough training data. For this question, we have conducted experiments on a full training data setting (Section 5.2), which considers complete training data. Furthermore, this section compares the results of the proposed model against the results generated by state-of-the-art baseline models on Frames and key–value retrieval datasets.

5.1. Learning from limited training data

Limited training data setting results are presented in Table 4. From the table, it can be observed that contextual models work better than the model with no context i.e Att-ED, or model with just previous turn context i.e. PrevTurn. Furthermore, results reveal that dialogue context is essential for multi-turn intent determination and slot filling tasks, giving an improvement of 4% and 5% F1 respectively on the KVRET dataset, and 7% and 6% improvement on the Frames dataset.

Furthermore, it can be observed in Fig. 7 and Table 4 that DFFTA and TDA models achieved superior results on both KVRET and Frames dataset. Because, these models can pay more attention to important history utterance by automatically decaying weights over time and manually decay weights over time, respectively. While on the Frames dataset HRNN produces superior results in comparison to SDEN and MemNet, as it utilizes dialogue context vector efficiently by initializing the hidden states of RNN tagger from it, while the dialogue act vector is fed as an additional input to RNN. In contrast, SDEN and MemNet use dialogue context to initialize the hidden state of RNN or as an additional input to RNN tagger. Furthermore, jointBERT achieved superior results than MemNet, SDEN, DFFTA, and TDA models as it leverages contextual word information using a pre-trained BERT model as opposed to static word embedding used by former models. However, NN-RE outperforms jointBERT in both precision and F1 score metrics, as the model lack domain information about slot types and particular patterns it may contain. On the contrary, the NN-RE model provides additional information about the slot type and its value, which boosts the performance of the slot filling task.

The results in Fig. 8 reveals that HRNN produces worse results than SDEN and MemNet on KVRET datasets, as dialogue acts information is limited in the KVRET dataset, and the dialogue context vector is too weak. Besides, Fig. 9 shows that the MemNet model achieved superior results on the intent determination task as it utilizes a single RNN tagger for jointly generating intent and slot information. Furthermore, the DFFTA model produced the superior on the intent determination task as it applies the sentence-level attention mechanism by leveraging the trainable distance vector to exploit the temporal information of the dialogue context. However, our proposed multi-task NN model based on RNN and dilated CNN significantly improved the performance of the intent determination task by almost 1% in terms of F1 score without degrading slot filling performance. The reason behind the improvement is that dilated CNN enables our model to capture the most important ngram with a wider context instead of a sequential RNN model. Additionally, if we look at the results of NN, it shows that the NN model learns task-specific features in an effective manner which, boosts the overall performance of the model.

Furthermore, the proposed NN-RE model provides a substantial boost on the F1 score of both intent determination and slot filling tasks over both KVRET and Frames datasets. The reason behind NN-RE model superior performance is that slot REs, and intent REs encode domain information about slot types and intent type in the utterance, respectively. This additional information helps the neural network to generalize better on smaller training data. We also observe that REs have less impact on results performance when there are large training data, it is not surprising as knowledge encoded in REs can be captured by exploiting a large amount of training data.

5.2. Learning from full training data

The Full training data setting results are presented in Table 5. The results reveal that attention Encoder–Decoder performs poorly on both datasets against all baseline competitors, as it lacks in exploiting contextual information. Similarly, the Previous Turn context model generated the second poorest results, because the model merely considers previous turn context, ignoring the full dialogue information and therefore fail to determine the user intent and utterance slots. On the other hand, MemNet, SDEN, HRNN, TDA, DFFTA, NN models have exploited the full dialogue context; therefore, they perform better than the former models.

Results on two datasets reveal that the MemNet model produced superior results on the KVRET dataset as it utilizes dialogue context more effectively by concatenating it with LSTM input during each time step. In contrast, the SDEN model achieved superior results on the Frames dataset because initial states of the LSTM tagger of SDEN are generated from dialogue context, which allowed the tagger to better handle noise in the dialogue context. Comparing the results on two datasets indicate that SDEN worked better than MemNet when dialogue length grew.

Moreover, the HRNN model produces superior results on the Frames dataset and inferior results on the KVRET dataset; as it uses system dialogue acts to generate dialogue context which is partially available in the KVRET dataset. Furthermore, the DFFTA and TDA model achieved superior results on both datasets as compared to other contextual models, as these models exploit the temporal information of dialogue history. The results indicate that DFFTA performs better than TDA as it automatically learns flexible and optimal time-decay function by using distance embeddings instead of utilizing manual time decay used by the TDA model. Besides, the JointBERT model produces superior results than DFFTA and TDA models because it exploits the pre-trained language model trained on large-scale English corpora. However, it is evident from the results that the proposed NN-RE model significantly outperformed all baseline competitors, as it leverages contextual word embedding from BERT model which is utilized in task-specific CNN and RNN networks to learn local and temporal features while leveraging the shared dialogue context information for intent determination and slot filling tasks. Furthermore, our model exploits domain knowledge encoded through regular expressions, hence further improving the F1 score on both tasks.

Table 4
Learning from limited training data results for the proposed Neural Network without Regular expression (NN) and Neural Network with Regular expression (NN-RE) models against baseline methods.

Dataset	Model	Context	Intent			Slot		
			Precision	Recall	F1 Score	Precision	Recall	F1 Score
KVRET	Att-ED [4]	None	86.74	85.41	86.07	66.51	55.91	60.73
	PrevTurn	PrevTurn	87.06	86.56	86.80	60.84	62.33	61.58
	MemNet [8]	History	89.62	86.71	88.10	60.39	67.43	63.13
	SDEN [9]	History	88.44	87.39	87.91	67.13	68.67	62.27
	HRNN [7]	Dialogue acts	87.26	86.73	86.99	61.08	62.56	61.86
	TDA [10]	History	89.63	86.82	88.15	61.41	67.75	63.45
	DFFTA [63]	History	89.67	87.16	88.34	60.87	68.05	63.74
	JointBERT [64]	History	90.14	89.47	90.13	68.82	63.58	65.81
	NN	History	90.72	89.94	90.51	68.19	62.74	65.16
	NN-RE	History	92.83	91.44	**92.16**	70.61	64.37	**67.23**
Frames	Att-ED [4]	None	74.36	69.86	72.04	58.31	49.23	53.34
	PrevTurn	PrevTurn	75.80	72.77	74.25	54.31	58.06	55.65
	MemNet [8]	History	76.29	73.72	74.54	60.12	54.23	57.00
	SDEN [9]	History	77.87	74.72	76.27	56.87	59.32	58.05
	HRNN [7]	Dialogue acts	77.80	75.29	76.53	57.12	59.65	58.36
	TDA [10]	History	78.29	76.63	77.10	60.15	56.86	58.41
	DFFTA [63]	History	78.71	77.14	77.89	60.37	57.15	58.67
	JointBERT [64]	History	79.51	78.91	79.37	60.91	58.72	59.64
	NN	History	80.25	79.05	79.82	60.73	58.37	59.28
	NN-RE	History	82.26	79.74	**81.17**	62.85	57.64	**60.13**

Table 5
Learning from full training data results for multi-turn Intent Determination and slot filling.

Dataset	Model	Context	Intent			Slot		
			Precision	Recall	F1 Score	Precision	Recall	F1 Score
KVRET	Att-ED [4]	None	92.86	91.76	92.31	70.25	76.63	73.30
	PrevTurn	PrevTurn	94.58	93.62	94.10	70.60	78.03	74.13
	MemNet [8]	History	98.01	98.01	98.01	71.53	78.27	74.75
	SDEN [9]	History	97.83	98.18	97.99	71.31	78.09	74.54
	HRNN [7]	Dialogue acts	97.69	98.01	97.84	70.99	78.04	74.35
	TDA [10]	History	98.04	98.10	98.09	72.04	77.93	74.87
	DFFTA [63]	History	98.18	98.18	98.16	72.19	78.21	75.14
	JointBERT [64]	History	98.20	98.24	98.22	73.24	79.06	76.32
	NN	History	98.22	98.45	98.31	73.18	78.75	76.06
	NN-RE	History	98.50	98.46	**98.48**	74.32	79.18	**76.91**
Frames	Att-ED [4]	None	92.95	72.14	81.23	66.94	67.33	67.13
	PrevTurn	PrevTurn	88.35	87.14	87.74	67.51	71.49	69.44
	MemNet [8]	History	91.88	92.43	92.15	69.45	72.31	70.85
	SDEN [9]	History	92.98	91.61	92.29	70.70	71.98	71.33
	HRNN [7]	Dialogue acts	93.25	91.79	92.51	71.82	71.23	71.42
	TDA [10]	History	92.45	92.36	92.56	70.64	72.34	71.58
	DFFTA [63]	History	92.52	92.48	92.67	70.72	72.47	71.71
	JointBERT [64]	History	93.96	92.10	93.31	71.16	73.42	72.51
	NN	History	93.60	92.27	93.06	70.38	73.59	71.94
	NN-RE	History	95.32	92.95	**94.17**	74.23	73.15	**73.53**

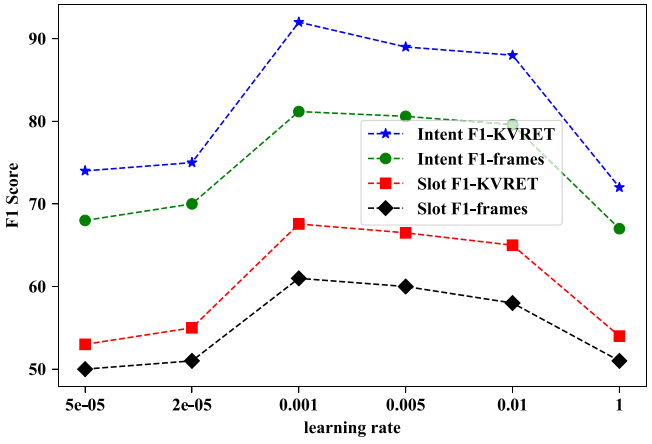

Fig. 10. Effect of learning rate.

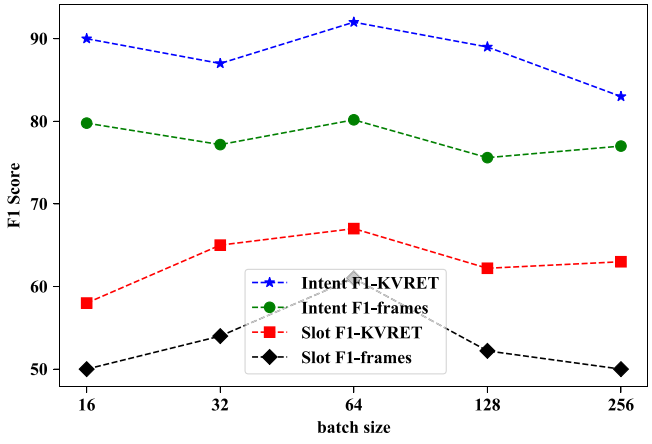

Fig. 11. Effect of batch size.

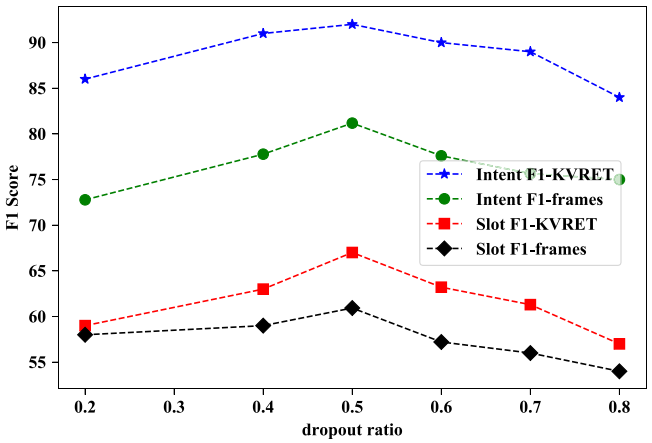

Fig. 12. Effect of dropout ratio.

5.3. Effect of hyper-parameters

Table 3 shows the hyper-parameters setting used in this research. The most important hyper-parameters of the proposed model are learning rate, batch size, dropout, hidden layers, and dilation factor. We analysed the performance of our model concerning these parameters. We use the random search [65] method to search hyper-parameters randomly from a specified subset of hyper-parameters. The learning rate for optimizing the model was chosen log-uniformly from 0.00005 to 1. In Fig. 10, we can observe that the model performs poorly on a learning rate of 1, which indicates that the model is unable to converge on a high learning rate. In contrast, using a too-small learning rate of 0.00005, the model takes more time to converge. On the KVRET dataset, the performance of the model increases steadily, especially on smaller learning rates of 0.001 and 0.005, and obtains F1 scores greater than 67 and 92 on the slot filling and intent detection, respectively. To choose an adequate learning rate, we apply a fine search scheme and find that our model generates the most significant results when configured on a learning rate of 0.001. Furthermore, we select batch sizes ranging from 2 to 1000. Fig. 11 reveals that in limited training dataset, the model obtained the best performance with smaller batch sizes such as 32, and 64. Whereas, the learning efficiency of our model decreased for the larger batch sizes i.e., 128, 256, 512. When batch size is set to 64, our model achieves the best F1 score on both datasets. That is close to 60 and 81 for slot filling and intent detection on the Frames dataset, while the model achieved an F1 score 67 and 92 for slot filling and intent detection respectively on the KVRET dataset. Therefore, we chose batch size 64 for the limited training data and full training data setting. Furthermore, we analyse the effect of the dropout layer depicted in Fig. 12. We observed a significant improvement for both intent determination and slot filling tasks when dropout is between 0.4 and 0.6. When dropout is greater than 0.6 model pays focus to the regular expression output and overlooks neural network output; therefore, it does not generalize well. For dilated convolution, the most important factor is the kernel size k and dilated factor d; we empirically found that $k = 3$ and $d = 4$ provide a sufficiently large receptive field for intent determination task. We also observed that the full training data was far less sensitive to hyper-parameters choice than the limited training data.

6. Conclusions

In this paper, we have proposed a model that combines the expressive power of REs with the generalization ability of the neural network for multi-turn intent determination and slot filling tasks. The proposed multi-task model obtained contextual word representations of user utterances from the pre-trained BERT model. These representations were used by memory networks to encode dialogue context which is shared by the tasks. Furthermore, dilated CNN and RNN networks have been employed on top of contextual word representations and dialogue context vector to learn task-specific features for intent determination and slot filling tasks. These neural networks have been combined with REs to encode domain knowledge about intent and slot values. Experiments on two real-world datasets demonstrate that the combination improves the performance of the model in both the full training data and limited training data setting. We showed that encoding domain knowledge into neural network training significantly improved the performance of neural networks. The encouraging results indicate that the proposed model can be used for improving other applications of text classification and sequence labelling.

In future work, we plan to extend our model for dialogue state tracking and natural language generation. A regular expression can be used to track the dialogue state and generate an appropriate response would be possible. Also, we aim to generalize our framework to automatically generate regular expressions from training data.

CRediT authorship contribution statement

Waheed Ahmed Abro: Conceptualization, Methodology, Software, Data curation. **Guilin Qi:** Supervision, Conceptualization. **Zafar Ali:** Writing - original draft, Writing - review & editing, Validation,. **Yansong Feng:** Conceptualization. **Muhammad Aamir:** Writing - review & editing, Visualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

Research presented in this paper was partially supported by the National Key Research and Development Program of China under grants (2018YFC0830200), the Natural Science Foundation of China grants (U1736204, 61602259), the Judicial Big Data Research Centre, School of Law at Southeast University, China, and the project no. 31511120201 and 31510040201.

References

[1] G. Tur, R. De Mori, Spoken Language Understanding: Systems for Extracting Semantic Information from Speech, John Wiley & Sons, 2011.
[2] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, et al., Using recurrent neural networks for slot filling in spoken language understanding, IEEE/ACM Trans. Audio Speech Lang. Process. 23 (3) (2015) 530–539.
[3] D. Hakkani-Tür, G. Tür, A. Celikyilmaz, Y.-N. Chen, J. Gao, L. Deng, Y.-Y. Wang, Multi-domain joint semantic frame parsing using bi-directional RNN-LSTM, in: INTERSPEECH, 2016, pp. 715–719.
[4] B. Liu, I. Lane, Attention-based recurrent neural network models for joint intent detection and slot filling, in: INTERSPEECH 2016, 2016, pp. 685–689, http://dx.doi.org/10.21437/Interspeech.2016-1352.
[5] C.-W. Goo, G. Gao, Y.-K. Hsu, C.-L. Huo, T.-C. Chen, K.-W. Hsu, Y.-N. Chen, Slot-gated modeling for joint slot filling and intent prediction, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 753–757, http://dx.doi.org/10.18653/v1/N18-2118, URL https://www.aclweb.org/anthology/N18-2118.

[6] A. Gupta, J. Hewitt, K. Kirchhoff, Simple, fast, accurate intent classification and slot labeling for goal-oriented dialogue systems, in: Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue, Association for Computational Linguistics, Stockholm, Sweden, 2019, pp. 46–55, http://dx.doi.org/10.18653/v1/W19-5906, URL https://www.aclweb.org/anthology/W19-5906.

[7] R. Gupta, A. Rastogi, D.Z. Hakkani, An efficient approach to encoding context for spoken language understanding, in: INTERSPEECH, 2018, pp. 3469–3473.

[8] Y.-N. Chen, D. Hakkani-Tür, G. Tür, J. Gao, L. Deng, End-to-end memory networks with knowledge carryover for multi-turn spoken language understanding, in: INTERSPEECH, 2016, pp. 3245–3249.

[9] A. Bapna, G. Tür, D. Hakkani-Tür, L. Heck, Sequential dialogue context modeling for spoken language understanding, in: Proceedings of the 18th Annual SIGDial Meeting on Discourse and Dialogue, Association for Computational Linguistics, Saarbrücken, Germany, 2017, pp. 103–114, http://dx.doi.org/10.18653/v1/W17-5514, URL https://www.aclweb.org/anthology/W17-5514.

[10] S.-Y. Su, P.-C. Yuan, Y.-N. Chen, How time matters: Learning time-decay attention for contextual spoken language understanding in dialogues, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 2133–2142, http://dx.doi.org/10.18653/v1/N18-1194, URL https://www.aclweb.org/anthology/N18-1194.

[11] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186, http://dx.doi.org/10.18653/v1/N19-1423, URL https://www.aclweb.org/anthology/N19-1423.

[12] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R.R. Salakhutdinov, Q.V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, in: Advances in Neural Information Processing Systems, 2019, pp. 5753–5763.

[13] J. Howard, S. Ruder, Universal language model fine-tuning for text classification, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 328–339, http://dx.doi.org/10.18653/v1/P18-1031, URL https://www.aclweb.org/anthology/P18-1031.

[14] S. Zhang, L. He, S. Vucetic, E. Dragut, Regular expression guided entity mention mining from noisy web data, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 1991–2000, http://dx.doi.org/10.18653/v1/D18-1224, URL https://www.aclweb.org/anthology/D18-1224.

[15] A.X. Chang, C.D. Manning, TokensRegex: Defining cascaded regular expressions over tokens, Tech. Rep. CSTR 2014-02, Department of Computer Science, Stanford University, 2014.

[16] W.A. Abro, G. Qi, H. Gao, M.A. Khan, Z. Ali, Multi-turn intent determination for goal-oriented dialogue systems, in: 2019 International Joint Conference on Neural Networks (IJCNN), 2019, pp. 1–8.

[17] R. Sarikaya, G.E. Hinton, B. Ramabhadran, Deep belief nets for natural language call-routing, in: International Confrence on Acoustics, Speech, and Signal Processing, IEEE, 2011, pp. 5680–5683.

[18] G. Tur, L. Deng, D. Hakkani-Tür, X. He, Towards deeper understanding: Deep convex networks for semantic utterance classification, in: International Confrence on Acoustics, Speech, and Signal Processing, IEEE, 2012, pp. 5045–5048.

[19] Z. Zhang, L. Luo, Hate speech detection: A solved problem? The challenging case of long tail on twitter, Semant. Web 10 (2018) 925–945.

[20] S.V. Ravuri, A. Stolcke, Recurrent neural network and LSTM models for lexical utterance classification, in: INTERSPEECH, ISCA, 2015, pp. 135–139.

[21] S. Ravuri, A. Stolcke, A comparative study of recurrent neural network models for lexical domain classification, in: International Confrence on Acoustics, Speech, and Signal Processing, IEEE, 2016, pp. 6075–6079.

[22] K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, D. Yu, Recurrent neural networks for language understanding., in: INTERSPEECH, 2013, pp. 2524–2528.

[23] L. Deng, D. Yu, Deep convex net: A scalable architecture for speech pattern classification, in: INTERSPEECH, 2011, pp. 2285–2288.

[24] T.-E. Lin, H. Xu, A post-processing method for detecting unknown intent of dialogue system via pre-trained deep neural network classifier, Knowl.-Based Syst. 186 (2019) 104979, http://dx.doi.org/10.1016/j.knosys.2019.104979, URL http://www.sciencedirect.com/science/article/pii/S0950705119304034.

[25] N. Howard, E. Cambria, Intention awareness: improving upon situation awareness in human-centric environments, Human-centric Comput. Inf. Sci. 3 (1) (2013) 1–17, http://dx.doi.org/10.1186/2192-1962-3-9.

[26] R. Liu, X. Zhang, J. Webb, S. Li, Context-specific intention awareness through web query in robotic caregiving, in: 2015 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2015, pp. 1962–1967.

[27] R. López-Cózar, Using knowledge on word-islands to improve the performance of spoken dialogue systems, Knowl.-Based Syst. 88 (2015) 223–243, http://dx.doi.org/10.1016/j.knosys.2015.07.029, URL http://www.sciencedirect.com/science/article/pii/S095070511500283X.

[28] Y.-B. Kim, S. Lee, R. Sarikaya, Speaker-sensitive dual memory networks for multi-turn slot tagging, in: 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 2017, pp. 541–546.

[29] M. Firdaus, A. Kumar, A. Ekbal, P. Bhattacharyya, A multi-task hierarchical approach for intent detection and slot filling, Knowl.-Based Syst. 183 (2019) 104846, http://dx.doi.org/10.1016/j.knosys.2019.07.017, URL http://www.sciencedirect.com/science/article/pii/S0950705119303211.

[30] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 2227–2237, http://dx.doi.org/10.18653/v1/N18-1202, URL https://www.aclweb.org/anthology/N18-1202.

[31] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, Improving language understanding by generative pre-training (2018), 2018, URL https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf.

[32] L. Dong, N. Yang, W. Wang, F. Wei, X. Liu, Y. Wang, J. Gao, M. Zhou, H.-W. Hon, Unified language model pre-training for natural language understanding and generation, in: Advances in Neural Information Processing Systems, 2019, pp. 13063–13075.

[33] X. Liu, P. He, W. Chen, J. Gao, Multi-task deep neural networks for natural language understanding, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019, pp. 4487–4496, http://dx.doi.org/10.18653/v1/P19-1441, URL https://www.aclweb.org/anthology/P19-1441.

[34] Y. Sun, S. Wang, Y. Li, S. Feng, X. Chen, H. Zhang, X. Tian, D. Zhu, H. Tian, H. Wu, Ernie: Enhanced representation through knowledge integration, 2019, arXiv preprint arXiv:1904.09223.

[35] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, S. Bowman, GLUE: A multi-task benchmark and analysis platform for natural language understanding, in: Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 353–355, http://dx.doi.org/10.18653/v1/W18-5446, URL https://www.aclweb.org/anthology/W18-5446.

[36] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, S. Bowman, Superglue: A stickier benchmark for general-purpose language understanding systems, in: Advances in Neural Information Processing Systems, 2019, pp. 3266–3280.

[37] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, J. Gao, Deep learning based text classification: A comprehensive review, 2020, arXiv preprint arXiv:2004.03705.

[38] M.E. Peters, S. Ruder, N.A. Smith, To tune or not to tune? Adapting pretrained representations to diverse tasks, in: Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019), Association for Computational Linguistics, Florence, Italy, 2019, pp. 7–14, http://dx.doi.org/10.18653/v1/W19-4302, URL https://www.aclweb.org/anthology/W19-4302.

[39] Z. Hu, X. Ma, Z. Liu, E. Hovy, E. Xing, Harnessing deep neural networks with logic rules, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Berlin, Germany, 2016, pp. 2410–2420, http://dx.doi.org/10.18653/v1/P16-1228, URL https://www.aclweb.org/anthology/P16-1228.

[40] T. Alashkar, S. Jiang, S. Wang, Y. Fu, Examples-Rules Guided Deep Neural Network for Makeup Recommendation, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, 2017, pp. 941–947.

[41] B. Luo, Y. Feng, Z. Wang, S. Huang, R. Yan, D. Zhao, Marrying up regular expressions with neural networks: A case study for spoken language understanding, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 2083–2093, http://dx.doi.org/10.18653/v1/P18-1194, URL https://www.aclweb.org/anthology/P18-1194.

[42] S. Guo, Q. Wang, L. Wang, B. Wang, L. Guo, Knowledge graph embedding with iterative guidance from soft rules, in: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, 2018, pp. 4816–4823.

[43] H. Zhou, T. Young, M. Huang, H. Zhao, J. Xu, X. Zhu, Commonsense knowledge aware conversation generation with graph attention, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, AAAI Press, 2018, pp. 4623–4629.

[44] J. Guan, Y. Wang, M. Huang, Story ending generation with incremental encoding and commonsense knowledge, in: Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 6473–6480.
[45] T. Young, V. Pandelea, S. Poria, E. Cambria, Dialogue systems with audio context, Neurocomputing 388 (2020) 102–109, http://dx.doi.org/10.1016/j.neucom.2019.12.126, URL http://www.sciencedirect.com/science/article/pii/S0925231220300758.
[46] T.-H. Wen, Y. Miao, P. Blunsom, S. Young, Latent intention dialogue models, in: Proceedings of the 34th International Conference on Machine Learning - Volume 70, ICML'17, JMLR.org, 2017, pp. 3732–3741.
[47] H. Xu, H. Peng, H. Xie, E. Cambria, L. Zhou, W. Zheng, End-to-end latent-variable task-oriented dialogue system with exact log-likelihood optimization, World Wide Web 23 (3) (2020) 1989–2002, http://dx.doi.org/10.1007/s11280-019-00688-8.
[48] M. Alirezaie, M. Längkvist, M. Sioutis, A. Loutfi, Semantic referee: A neural-symbolic framework for enhancing geospatial semantic segmentation, Semant. Web 10 (2019) 863–880, http://dx.doi.org/10.3233/SW-190362.
[49] N. Locascio, K. Narasimhan, E. DeLeon, N. Kushman, R. Barzilay, Neural generation of regular expressions from natural language with minimal domain knowledge, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Austin, Texas, 2016, pp. 1918–1923, http://dx.doi.org/10.18653/v1/D16-1197, URL https://www.aclweb.org/anthology/D16-1197.
[50] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.
[51] Y. Wu, M. Schuster, Z. Chen, Q.V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, L. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G.S. Corrado, M. Hughes, J. Dean, Google's neural machine translation system: Bridging the gap between human and machine translation, 2016, CoRR abs/1609.08144.
[52] S. Bai, J.Z. Kolter, V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling., CoRR abs/1803.01271 (2018).
[53] A. Jacovi, O. Sar Shalom, Y. Goldberg, Understanding convolutional neural networks for text classification, in: Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 56–65, http://dx.doi.org/10.18653/v1/W18-5408, URL https://www.aclweb.org/anthology/W18-5408.
[54] J.L. Elman, Finding structure in time, Cogn. Sci. 14 (2) (1990) 179–211, http://dx.doi.org/10.1016/0364-0213(90)90002-E, URL http://www.sciencedirect.com/science/article/pii/036402139090002E.
[55] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780, http://dx.doi.org/10.1162/neco.1997.9.8.1735.
[56] Y. Li, R. Krishnamurthy, S. Raghavan, S. Vaithyanathan, H.V. Jagadish, Regular expression learning for information extraction, in: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Honolulu, Hawaii, 2008, pp. 21–30, URL https://www.aclweb.org/anthology/D08-1003.
[57] E. Strubell, P. Verga, D. Belanger, A. McCallum, Fast and accurate entity recognition with iterated dilated convolutions, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Copenhagen, Denmark, 2017, pp. 2670–2680, http://dx.doi.org/10.18653/v1/D17-1283, URL https://www.aclweb.org/anthology/D17-1283.
[58] C.T. Hemphill, J.J. Godfrey, G.R. Doddington, The ATIS spoken language systems pilot corpus, in: Speech and Natural Language: Proceedings of a Workshop Held At Hidden Valley, Pennsylvania, June 24-27, 1990, 1990, pp. 24–27.
[59] L. El Asri, H. Schulz, S. Sharma, J. Zumer, J. Harris, E. Fine, R. Mehrotra, K. Suleman, Frames: a corpus for adding memory to goal-oriented dialogue systems, in: Proceedings of the 18th Annual SIGDial Meeting on Discourse and Dialogue, Association for Computational Linguistics, Saarbrücken, Germany, 2017, pp. 207–219, http://dx.doi.org/10.18653/v1/W17-5526, URL https://www.aclweb.org/anthology/W17-5526.
[60] M. Eric, L. Krishnan, F. Charette, C.D. Manning, Key-value retrieval networks for task-oriented dialogue, in: Proceedings of the 18th Annual SIGDial Meeting on Discourse and Dialogue, Association for Computational Linguistics, Saarbrücken, Germany, 2017, pp. 37–49, http://dx.doi.org/10.18653/v1/W17-5506, URL https://www.aclweb.org/anthology/W17-5506.
[61] J.E. Friedl, Mastering Regular Expressions, O'Reilly Media, Inc., 2006.
[62] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, CoRR abs/1412.6980.
[63] J. Kim, J.-H. Lee, Decay-function-free time-aware attention to context and speaker indicator for spoken language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 3718–3726, http://dx.doi.org/10.18653/v1/N19-1372, URL https://www.aclweb.org/anthology/N19-1372.
[64] Q. Chen, Z. Zhuo, W. Wang, BERT For joint intent classification and slot filling, 2019, CoRR abs/1902.10909.
[65] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, J. Mach. Learn. Res. 13 (Feb) (2012) 281–305.