Improving Response Selection in Multi-turn Dialogue Systems by Incorporating Domain Knowledge

Debanjan Chaudhuri
Smart Data Analytics Group
University of Bonn & Fraunhofer IAIS
Germany
chaudhur@cs.uni-bonn.de

Agustinus Kristiadi
Smart Data Analytics Group
University of Bonn
Germany
kristiadi@uni-bonn.de

Jens Lehmann
Smart Data Analytics Group
University of Bonn & Fraunhofer IAIS
Germany
jens.lehmann@cs.uni-bonn.de

Asja Fischer
Department of Mathematics
Ruhr University Bochum
Germany
asja.fischer@rub.de

Abstract

Building systems that can communicate with humans is a core problem in Artificial Intelligence. This work proposes a novel neural network architecture for response selection in an end-to-end multi-turn conversational dialogue setting. The architecture applies context level attention and incorporates additional external knowledge provided by descriptions of domain-specific words. It uses a bi-directional Gated Recurrent Unit (GRU) for encoding context and responses and learns to attend over the context words given the latent response representation and vice versa. In addition, it incorporates external domain specific information using another GRU for encoding the domain keyword descriptions. This allows better representation of domain-specific keywords in responses and hence improves the overall performance. Experimental results show that our model outperforms all other state-of-the-art methods for response selection in multi-turn conversations.

<table>
<tr><th>Context</th></tr>
<tr><td><b>Utterance 1:</b>
My networking card is not working on my Ubuntu, can somebody help me?
<b>Utterance 2:</b>
What's your kernel version? Run <i>uname</i> -r or <i>sudo dpkg</i> -l |<i>grep</i> linux-headers |<i>grep</i> ii |<i>awk</i> '{<i>print</i> $3}' and paste the output here.
<b>Utterance 3:</b>
It's 2.8.0-30-generic.
<b>Utterance 4:</b>
Your card is not supported in that kernel. You need to upgrade, that's like decade old kernel!
<b>Utterance 5:</b>
Ok how do I install the new kernel??</td></tr>
<tr><th>Response</th></tr>
<tr><td>Just do <i>sudo apt-get</i> upgrade, that's it.</td></tr>
</table>

Table 1: Illustration of a multi-turn conversation with domain specific words (UNIX commands) in italics.

1 Introduction

In a conversation scenario, a dialogue system can be applied to the task of freely generating a new response or to the task of selecting a response from a set of candidate responses based on the previous utterances, i.e. the context of the dialogue. The former is known as *generative* dialogue system while the latter is called *retrieval-based* (or response selection) dialogue system.

Both approaches can be realized using a modular architecture, where each module is responsible for a certain task such as natural language understanding, dialogue state-tracking, natural language generation, etc., or can be trained in an end-to-end manner optimized on a single objective function.

Previous work, belonging to the latter category, by Lowe et al. (2015a) applied neural networks to multi-turn response selection in conversations by encoding the utterances in the context as well as the possible responses with a Long Short-term Memory (LSTM) (Hochreiter and Schmidhuber, 1997). Based on the context and response encodings, the neural network then estimates the probability for each response to be the correct one given the context. More recently, a lot of enhanced architectures have been proposed that build on the

Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL 2018), pages 497–507
Brussels, Belgium, October 31 - November 1, 2018. ©2018 Association for Computational Linguistics

general idea of encoding response and context first and performing some embedding-based matching after (Yan et al., 2016; Zhou et al., 2016; An et al., 2018; Dong and Huang, 2018).

Although such approaches result in efficient text-pair matching capabilities, they fail to attend over logical consistencies for longer utterances in the context, given the response. Moreover, in domain specific scenarios, a system's ability to incorporate additional domain knowledge can be very beneficial, e.g. for the example shown in Table 1.

In this paper, we propose a novel neural network architecture for multi-turn response-selection that extends the model proposed by Lowe et al. (2015a). Our major contributions are: (1) a neural network paradigm that is able to attend over important words in a context utterance given the response encoding (and vice versa), (2) an approach to incorporate additional domain knowledge into the neural network by encoding the description of domain specific words with a GRU and using a bilinear operation to merge the resulting domain specific representations with the vanilla word embeddings, and (3) an empirical evaluation on a publicly available multi-turn dialogue corpus showing that our system outperforms all other state-of-the-art methods for response selection in a multi-turn setting.

2 Related work

Recently, human-computer conversations have attracted increasing attention in the research community and dialogue systems have become a field of research on its own. The conversation models proposed in early studies (Walker et al., 2001; Oliver and White, 2004; Stent et al., 2002) were designed for catering to specific domains only, e.g. for performing restaurant bookings, and required substantial rule-based strategy building and human efforts in the building process. With the advancements in machine learning, there have been more and more studies on conversational agents which are based on data-driven approaches. Data-driven dialogue systems can chiefly be realized by two types of architectures: (1) pipeline architectures, which follow a modular pattern for modelling the dialogues, where each component is trained/created separately to perform a specific sub-task, and (2) end-to-end architectures, which consist of a single trainable module for modelling the conversations.

Task-oriented dialogue systems, which are designed to assist users in achieving specific goals, were mainly realized by pipeline architectures. Recently however, there have been more and more works on end-to-end dialogue systems because of the limitations of the former modular architectures, namely, the credit assignment problem and inter-component dependency, as for example described by Zhao and Eskenazi (2016). Wen et al. (2017) and Bordes et al. (2017) proposed encoder-decoder-based neural networks for modeling task oriented dialogues. Moreover, Zhao and Eskenazi (2016) proposed an end-to-end reinforcement learning-based system for jointly learning to perform dialogue state-tracking (Williams et al., 2013) and policy learning (Baird, 1995).

Since task oriented systems primarily focus on completing a specific task, they usually do not allow free flowing, articulate conversations with the user. Therefore, there has been considerable effort to develop non-goal driven dialogue systems, which are able to converse with humans on an open domain (Ritter et al., 2011). Such systems can be modeled using either generative architectures, which are able to freely generate responses to user queries, or retrieval-based systems, which pick a response suitable to a context utterance out of a provided set of responses. Retrieval-based systems are therefore more limited in their output while having the advantage of producing more informative, constrained, and grammatically correct responses (Ji et al., 2014).

2.1 Generative models

Ritter et al. (2011) were the first to formulate the task of automatic response generation as phrase-based statistical machine translation, which they tackled with n-gram-based language models. Later approaches (Shang et al., 2015; Vinyals and Le, 2015; Luong et al., 2015) applied Recurrent Neural Network (RNN)-based encoder-decoder architectures. However, dialogue generation is considerably more difficult than language translation because of the wide possibility of responses in interactions. Also, for dialogues, in order to generate a suitable response at a certain time-step, knowing only the previous utterance is often not enough and the ability to leverage the context from the sequence of previous utterances is required. To overcome such challenges, a hierarchical RNN encoder-decoder-based system has been proposed by Serban et al. (2016) for leveraging contextual information in conversations.

2.2 Retrieval-based models

Earlier works on retrieval-based systems focused on modeling short-text, single-turn dialogues. Hao et al. (2013) introduced a data set for this task and proposed a response selection system which is based on information retrieval techniques like the vector space model and semantic matching. Ji et al. (2014) suggested to apply a deep neural network for matching contexts and responses, while Wu et al. (2016) proposed a topic aware convolutional neural tensor network for answer retrieval in short-text scenarios.

More recently, there has been a lot of focus on developing retrieval-based models for multi-turn dialogues which is more challenging as the models need to take into account long-term dependencies in the context. Lowe et al. (2015a), introduced the Ubuntu Dialogue Corpus (UDC), which is the largest freely available multi-turn dialogue data set. Moreover, the authors proposed to leverage RNNs, e.g. LSTMs, to encode both the context and the response, before computing the score of the pair based on the similarity of the encodings (w.r.t. a certain measure). This class of methods is referred to as dual encoder architectures. Shortly after, Kadlec et al. (2015) investigated the performance of dual encoders with different kind of encoder networks, such as convolutional neural networks (CNNs) and bi-directional LSTMs. Yan et al. (2016) followed a different approach and trained a single CNN to map a context-response pair to the corresponding matching score.

Later on, various extensions of the dual encoder architecture have been proposed. Zhou et al. (2016) employed two encoders in parallel, one working on word- the other on utterance-level. Wu et al. (2017) proposed the Sequential Matching Network (SMN), where the candidate response is matched with every utterance in the context separately, based on which a final score is computed. The Cross Convolution Network (CNN) (An et al., 2018) extends the dual encoder with a cross convolution operation. The latter is a dot product between the embeddings of the context and response followed by a max-pooling operation. Both of the outputs are concatenated and fed into a fully-connected layer for similarity matching. Moreover, An et al. (2018) improve the representation of rare words by learning different embeddings for them from the data. Handling rare words has also been studied by Dong and Huang (2018), who proposed to handle Out-of-Vocabulary (OOV) words by using both pre-trained word embeddings and embeddings from task-specific data.

Furthermore, many models targeting response selection along with other sentence pair scoring tasks such as paraphrasing, semantic text scoring, and recognizing textual entailment have been proposed. Baudiš et al. (2016) investigated a stacked RNN-CNN architecture and attention-based models for sentence-pair scoring. Match-SRNN (Wan et al., 2016) employs a spatial RNN to capture local interactions between sentence pairs. Match-LSTM (Wang and Jiang, 2016) improves its matching performance by using LSTM-based, attention-weighted sentence representations. QA-LSTM (Tan et al., 2016) uses a simple attention mechanism and combines the LSTM encoder with a CNN.

Incorporating unstructured domain knowledge into dialogue system has initially been studied by Lowe et al. (2015b) and followed by Xu et al. (2016), who incorporated a loosely-structured knowledge base into a neural network using a special gating mechanism. They created the knowledge base from domain-specific data, however their model is not able to leverage any external domain knowledge.

3 Background

In this section, we will explain the task at hand and give a brief introduction to the neural network architectures our proposed model is based on.

3.1 Problem definition

Let the data set $\mathcal{D} = \{(c_i, r_i, y_i)\}_{i=1}^{M}$ be a set of M triples consisting of context c_i, response r_i, and ground truth label y_i. Each context is a sequence of utterances, that is $c_i = \{u_{il}\}_{l=1}^{L}$, where L is the maximum context length. We define an utterance as a sequence of words $\{w_t\}_{t=1}^{T}$. Thus, c_i can also be viewed as a sequence of words by concatenating all utterances in c_i. Each response r_i is an utterance and $y_i \in \{0, 1\}$ is the corresponding label of the given triple which takes a value of 1 if r_i is the correct response for c_i and 0 otherwise. The goal of retrieval-based dialogue systems is then to learn a predictive distribution $p(y|c, r, \boldsymbol{\theta})$ parameterized by $\boldsymbol{\theta}$. That is, given a context c and response r, we would like to infer the probability of r being a response to context c.

3.2 RNNs, BiRNNs and GRUs

Recurrent neural networks are one of the most popular classes of models for processing sequences of words $W = \{w_t\}_{t=1}^{T}$ with arbitrary length $T \in \mathbb{N}$, e.g. utterances or sentences. Each word w_t is first mapped onto its vector representation $\mathbf{w}_t$ (also referred to as word embedding), which serves as input to the RNN at time step t. The central element of RNNs is the recurrence relation of its hidden units, described by

$$\overrightarrow{\mathbf{h}}_t = f(\overrightarrow{\mathbf{h}}_{t-1}, \mathbf{w}_t|\boldsymbol{\phi}) \ , \tag{1}$$

where $\boldsymbol{\phi}$ are the parameters of the RNN and f is some nonlinear function. Accordingly, the state $\overrightarrow{\mathbf{h}}_t$ of the hidden units at time step t depends on the state $\overrightarrow{\mathbf{h}}_{t-1}$ in the previous time step and the t-th word in the sequence. This way, the hidden state $\overrightarrow{\mathbf{h}}_T$ obtained after T updates contains information about the whole sequence W, and can thus be regarded as an embedding of the sequence.

The RNN architecture can also be altered to take into account dependencies coming from both the past and the future by adding an additional sub-RNN that moves backward in time, giving rise to the name bi-directional RNN (biRNN). To achieve this, the network architecture is extended by an additional set of hidden units. The states $\overleftarrow{\mathbf{h}}_t$ of those hidden units are updated based on the current input word and the hidden state from the next time step. That is for $t = 1, \ldots, T-1$:

$$\overleftarrow{\mathbf{h}}_{T-t} = f(\overleftarrow{\mathbf{h}}_{T-t+1}, \mathbf{w}_{T-t}|\boldsymbol{\phi}) \ . \tag{2}$$

Here, the words are processed in reverse order, i.e. $w_T, \ldots, w_1$, such that $\overleftarrow{\mathbf{h}}_T$ (analogous to $\overrightarrow{\mathbf{h}}_T$ in the forward directed RNN) contains information about the whole sequence. At the t-th time step, the model's hidden representation of the sequence is then usually obtained by the concatenation of the hidden states from the forward and the backward RNN, i.e. by $\mathbf{h}_t = [\overrightarrow{\mathbf{h}}_t, \overleftarrow{\mathbf{h}}_t]$ and the embedding of the whole sequence W is given by $\mathbf{h}^W = [\overrightarrow{\mathbf{h}}_T, \overleftarrow{\mathbf{h}}_T]$.

Modeling very long sequences with RNNs is hard: Bengio et al. (1994) showed that RNNs suffer from vanishing and exploding gradients, which makes training over long-term dependency difficult. Such problems can be addressed by augmenting the RNN with additional gating mechanisms, as it is done in LSTMs and the Gated Recurrent Unit (GRU) (Cho et al., 2014). These mechanisms allow the RNN to learn how much to update the hidden state flexibly in each step and help the RNN to deal with the vanishing gradient problem in long sequences better than vanilla RNNs. The gating mechanism of GRUs is motivated by that of LSTMs, but is much simpler to compute and implement. It contains two gates, namely the reset and update gate, whose states at time t are denoted by $\mathbf{z_t}$ and $\mathbf{r_t}$, respectively. Formally, a GRU is defined by the following update equations

$$\begin{aligned}
\mathbf{z}_t &= \sigma(\mathbf{W}_z\mathbf{x}_t + \mathbf{U}_z\mathbf{h}_{t-1}) \ , \\
\mathbf{r}_t &= \sigma(\mathbf{W}_r\mathbf{x}_t + \mathbf{U}_r\mathbf{h}_{t-1}) \ , \\
\tilde{\mathbf{h}}_t &= \tanh(\mathbf{W}_h\mathbf{x}_t + \mathbf{U}_h\mathbf{r}_i \odot \mathbf{h}_{t-1}) \ , \\
\mathbf{h}_t &= \mathbf{z}_t \odot \tilde{\mathbf{h}}_t + (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} \ ,
\end{aligned}$$

where $\mathbf{x}_t$ is the input (corresponding to $\mathbf{w}_t$ in our setting) and the set of weight matrices $\phi = \{\mathbf{W}_z, \mathbf{U}_z, \mathbf{W}_r, \mathbf{U}_r, \mathbf{W}_h, \mathbf{U}_h\}$ constitute the learnable model parameters.

3.3 Dual Encoder

Recurrent neural networks and their variants have been used in many applications in the field of natural language processing, including retrieval-based dialogue systems. In this area the dual encoder (DE) (Lowe et al., 2015a) became a popular model. It uses a single RNN encoder to transform both context and response into low dimensional vectors and computes their similarity. More formally, let $\mathbf{h^c}$ and $\mathbf{h^r}$ be the encoded context and response, respectively. The probability of r being the correct response for c is then computed by the DE as

$$p(y|c, r, \boldsymbol{\theta}) = \sigma((\mathbf{h^c})^{\mathrm{T}}\, \mathbf{M}\, \mathbf{h^r} + b) \ , \tag{3}$$

where $\boldsymbol{\theta} = \{\boldsymbol{\phi}, \mathbf{M}, b\}$ (recall, that $\boldsymbol{\phi}$ is the set of parameters of the encoder RNN that outputs $\mathbf{h^c}$ and $\mathbf{h^r}$) is the set of parameters of the full model and σ is the sigmoid function. Note, that the same RNN is used to encode both context and response.

In summary, this approach can be described as first creating latent representations of context and response in the same vector space and then using the similarity between these latent embeddings (as induced by matrix $\mathbf{M}$ and bias b) for estimating the probability of the the response being the correct one for the given context.

4 Model description

Our model extends the DE described in Section 3.3 by two attention mechanisms which make the context encoding response-aware and vice versa. Furthermore, we augment the model with a mechanism for incorporating external knowledge to improve the handling of rare words. Both extensions are described in detail in the following subsections.

4.1 Attention augmented encoding

As described above, in the DE context and response are encoded independently from each other based on the same RNN. Instead of simply taking the final hidden state $\mathbf{h}^c$ (and $\mathbf{h}^r$) of the RNN as context (and response) encoding, we propose to use a response-aware attention mechanism to calculate the context embedding and vice versa.

Subsequently, we will describe this mechanism formally. Recall that a context c can be seen as sequence of words $\{w_t^c\}_{t=1}^T$ where all utterances are concatenated and T is the total number of words in the context. Given this sequence, the RNN (in our experiments a bi-directional GRU) produces a sequence of hidden states $\mathbf{h}_1^c, \ldots, \mathbf{h}_T^c$ and an encoding of the whole context sequence $\mathbf{h}^c$ as described in Section 3.2. Analogously, we get $\mathbf{h}_1^r, \ldots, \mathbf{h}_{T'}^r$ and $\mathbf{h}^r$ for a response consisting of a sequence of words $\{w_t^r\}_{t=1}^{T'}$, where T' is the total number of words in the response.

For calculating the response-aware context encoding, we first estimate attention weights α_t^c for the hidden state $\mathbf{h}_t^c$ in each time step, depending on the response encoding $\mathbf{h}^r$:

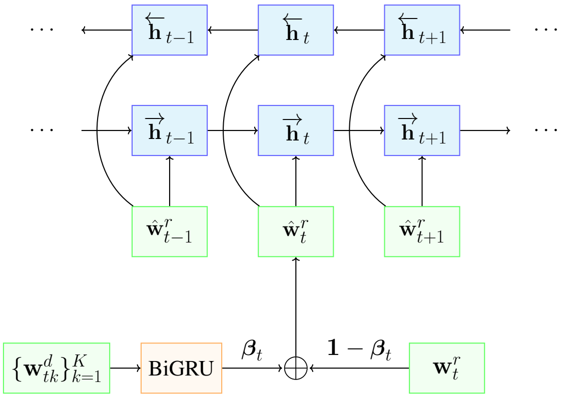
$$\alpha_t^c \propto \exp((\mathbf{h}_t^c)^\mathrm{T} \mathbf{W}_c \mathbf{h}^r) \ , \tag{4}$$

where $\mathbf{W}_c$ is a learnable parameter matrix. The response-aware context embedding then is given by

$$\hat{\mathbf{h}}^c = \sum_{t=1}^{T} \alpha_t^c \, \mathbf{h}_t^c \ . \tag{5}$$

Intuitively this means, that depending on the response we focus on different parts of the context sequence, for judging on how well the response matches the context. This may resemble human focus.

Similarly, we calculate the context-aware response encoding by



$\cdots$
$\overleftarrow{\mathbf{h}}_{t-1}$
$\overleftarrow{\mathbf{h}}_{t}$
$\overleftarrow{\mathbf{h}}_{t+1}$
$\cdots$
$\cdots$
$\overrightarrow{\mathbf{h}}_{t-1}$
$\overrightarrow{\mathbf{h}}_{t}$
$\overrightarrow{\mathbf{h}}_{t+1}$
$\cdots$
$\hat{\mathbf{w}}_{t-1}^r$
$\hat{\mathbf{w}}_{t}^r$
$\hat{\mathbf{w}}_{t+1}^r$
$\{\mathbf{w}_{tk}^d\}_{k=1}^K$
BiGRU
$\boldsymbol{\beta}_t$
$\mathbf{1} - \boldsymbol{\beta}_t$
$\mathbf{w}_t^r$

Figure 1: Our proposed way to incorporate domain knowledge into the model. $\boldsymbol{\beta}_t$ and $\mathbf{1} - \boldsymbol{\beta}_t$ represent the (multiplicative) weights for the description embedding and the word embedding respectively. The resulting combination, $\hat{\mathbf{w}}_t^r$ acts as an input of the encoder.

$$\hat{\mathbf{h}}^r = \sum_{t=1}^{T} \alpha_t^r \, \mathbf{h}_t^r \ , \tag{6}$$

with attention weights

$$\alpha_t^r \propto \exp((\mathbf{h}_t^r)^\mathrm{T} \mathbf{W}_r \mathbf{h}^c) \ . \tag{7}$$

The two attention-weighted encodings (for response and context, respectively) then replace the vanilla encodings in equation (3), that is

$$p(y|c, r, \boldsymbol{\theta}) = \sigma((\hat{\mathbf{h}}^c)^\mathrm{T} \, \mathbf{M} \, \hat{\mathbf{h}}^r + b) \ . \tag{8}$$

4.2 Incorporating domain keyword descriptions

Bahdanau et al. (2018) proposed a method for learning embeddings for OOV words based on external dictionary definitions. They learn these description embeddings of words using an LSTM for encoding the corresponding definition. If a particular word included in the dictionary also appears in the corpus' vocabulary (for which vanilla word embeddings are given), they add the word embedding and the description embedding together. Otherwise, in the case of OOV words, they use solely the description embedding in place of the missing word embedding. Inspired by this approach, we use a similar technique to incorporate domain keyword descriptions into word embeddings.

If a word w_t^r in the response utterance is in the set of domain keywords $\mathcal{K}$, we firstly extract its description. The description of w_t^r is a sequence of words $\{w_{tk}^d\}_{k=1}^K$, which is projected onto sequence of embeddings $\{\mathbf{w}_{tk}^d\}_{k=1}^K$. This sequence is encoded using another bi-directional GRU to obtain a vector representation $\mathbf{h}_t^d$ of the same dimension as the vanilla word embeddings. If w_t^r is not in $\mathcal{K}$, we simply set $\mathbf{h}_t^d$ to zero. We call $\mathbf{h}_t^d$ the *description embedding*.

Some domain specific words might also happen to be common words. For instance, in the case of the UDC's vocabulary, there exist tokens such as *shutdown* [1] or *who* [2], which are ambiguous, i.e., although they are valid UNIX commands, they are also common words in natural language. The description embeddings of domain specific words can be simply added to the vanilla word embeddings as suggested by Bahdanau et al. (2018). However, it might be advantageous if the model can determine itself whether to treat the current word as a domain specific word, a common word, or something in between, depending on the context. For instance, if the context is mainly talking about system users, then *who* is most likely a UNIX keyword. Therefore, we propose a more flexible way to combine the description embedding $\mathbf{h}_t^d$ and the word embedding $\mathbf{w}_t^r$, that is, we define the final word embedding to be a convex combination of both, and let the combination coefficients be given by a function of $\mathbf{h}_t^d$ and the context embedding $\hat{\mathbf{h}}^c$. Intuitively, this allows the model to flexibly focus on the description or the vanilla embedding, in dependence on the context and the description. Formally, the combination coefficients $\boldsymbol{\beta}_t$ of t-th word in the response is given by

$$\boldsymbol{\beta}_t \propto \exp(\mathbf{U}^{\mathrm{T}}\hat{\mathbf{h}}^c + \mathbf{V}^{\mathrm{T}}\mathbf{w}_t^r) \ , \tag{9}$$

where $\mathbf{U}$ and $\mathbf{V}$ are learnable parameter matrices. Note that $\boldsymbol{\beta}_t$ is a vector of the same dimension as the embeddings. The final embedding of w_t^r (which serves as input to the response encoder) is then the weighted sum

$$\hat{\mathbf{w}}_t^r = \boldsymbol{\beta}_t \odot \mathbf{h}_t^d + (\mathbf{1} - \boldsymbol{\beta}_t) \odot \mathbf{w}_t^r \ , \tag{10}$$

where $\odot$ denotes the element wise multiplication.

5 Experiment

5.1 Ubuntu multi-turn dialogue corpus

Extending the work of Uthus and Aha (2013), Lowe et al. (2015a) introduced a version of the Ubuntu chat log conversations which is the largest publicly available multi-turn, dyadic, and domain-specific dialogue data set. The chats are extracted from Ubuntu related topic specific chat rooms in the Freenode Internet Relay Chat (IRC) network. Usually, experienced users address a problem of someone by suggesting a potential solution and a *name mention* of the addressed user. A conversation between a pair of users often stops when the problem has been solved. However, they might continue having a discussion which is not related to the topic.

A preprocessed version of the above corpus and the needed vocabulary are provided by Wu et al. (2017). The preprocessing consisted of replacing numbers, URLs, and system paths with special placeholders as suggested by Xu et al. (2016). No additional preprocessing is performed by us. The data set consists of 1 million training triples, 500k validation triples, and 500k test triples. One half of the 1 million training triples are positive (triples with $y = 1$, i.e. the provided response fits the context) the other half negative (triples with $y = 0$). In contrast, in the validation and test set, for every context c_i, there exists one positive triple providing the ground-truth response to c_i and nine negative triples with unbefitting responses. Thus, in these sets, the ratio between positive and negative triples per context is 1:9 which makes evaluating the model with information retrieval metrics such as Recall@k possible (see Section 6).

5.2 Model hyperparameters

We chose a word embedding dimension of 200 as done by Wu et al. (2017). We use fastText (Bojanowski et al., 2016) to pre-train the word embeddings using the training set instead of using off-the-shelf word embeddings, following Wu et al. (2017). We set the hidden dimension of our GRU to be 300, as in the work of Lowe et al. (2015a). We restricted the sequence length of a context by a maximum of 320 words, and that of the response by 160. Because of the resulting size of the model and limited GPU memory, we had to use a smaller batch size of 32. We optimize the binary cross entropy loss of our model with respect to the training data using Adam (Kingma and Ba, 2015) with an initial learning rate of 0.0001. We train our model for a maximum of 20 epochs as according to our experience, this is more than enough to achieve convergence. The training is stopped when the validation recall does not increase after three sub-

[1] UNIX command for system shutdown.

[2] UNIX command to get a list of currently logged-in users.

Model	R_2@1	R_{10}@1	R_{10}@3	R_{10}@5
DE-RNN (Kadlec et al., 2015)	0.768	0.403	0.547	0.819
DE-CNN (Kadlec et al., 2015)	0.848	0.549	0.684	0.896
DE-LSTM (Kadlec et al., 2015)	0.901	0.638	0.784	0.949
DE-BiLSTM (Kadlec et al., 2015)	0.895	0.630	0.780	0.944
MultiView (Zhou et al., 2016)	0.908	0.662	0.801	0.951
DL2R (Yan et al., 2016)	0.899	0.626	0.783	0.944
r-LSTM (Xu et al., 2016)	0.889	0.649	0.857	0.932
MV-LSTM (Wan et al., 2016)	0.906	0.653	0.804	0.946
Match-LSTM (Wang and Jiang, 2016)	0.904	0.653	0.799	0.944
QA-LSTM (Tan et al., 2016)	0.903	0.633	0.789	0.943
SMN_{dyn} (Wu et al., 2017)	0.926	0.726	0.847	0.961
CCN (An et al., 2018)	-	0.727	0.858	0.971
ESIM (Dong and Huang, 2018)	-	0.734	0.854	0.967
AK-DE-biGRU (Ours)	**0.933**	**0.747**	**0.868**	**0.972**

Table 2: Evaluation results of our models compared to various baselines on Ubuntu Dialogue Corpus.

sequent epochs. The test set is evaluated on the model with the best validation recall.

For the implementation, we use PyTorch (Paszke et al., 2017). We train the model end-to-end with a single 12GB GPU. The implementation[3] of our models along with the additional domain knowledge base[4] are publicly available.

6 Results

Following Lowe et al. (2015a) and Kadlec et al. (2015), we use the Recall@k evaluation metric, where $R_n@k$ corresponds to the fraction of of examples for which the correct response is under the k best out of a set of n candidate responses, which were ranked according to there their probabilities under the model.

In our evaluation specifically, we use $R_2@1$, $R_{10}@1$, $R_{10}@3$, and $R_{10}@5$.

6.1 Comparison against baselines

We compare our model, which we refer to as *Attention and external Knowledge augmented DE with bi-directional GRU* (**AK-DE-biGRU**), against models previously tested on the same data set: the basic DE models analyzed by Lowe et al. (2015a) and Kadlec et al. (2015) using different encoders, such as convolutional neural network (**DE-CNN**), LSTM (**DE-LSTM**) and bi-directional LSTM (**DE-BiLSTM**); the **MultiView**, **DL2R** and **r-LSTM** models proposed by Zhou et al. (2016), Yan et al. (2016) and Xu et al. (2016), respectively; architectures for advanced context and response matching, namely **MV-LSTM** (Wan et al., 2016), **Match-LSTM** (Wang and Jiang, 2016), and **QA-LSTM** (Tan et al., 2016); architectures processing the context utterances individually, namely $\textbf{SMN}_{\textbf{dyn}}$ (Wu et al., 2017) and **CCN**; and we also use recently proposed **ESIM** (Dong and Huang, 2018) as a baseline.

The results are reported in Table 2. Our model outperforms all other models used as baselines. The largest improvement of our model compared to the best of the baselines (i.e. ESIM in general and SMN_{dyn} for $R_2@1$ metric) are with respect to the $R_{10}@1$ and $R_{10}@3$ metric, where we observed absolute improvements of 0.013 and 0.014 corresponding to 1.8% and 1.6% relative improvement , respectively. For $R_2@1$ and $R_{10}@5$ we observed more modest improvements of 0.007 (0.8%) and 0.005 (0.5%), respectively. Our results are significantly better with $p < 10^{-6}$ for a one-sample one-tailed t-test compared to the best baseline (ESIM), on $R_{10}@1$, $R_{10}@3$, $R_{10}@5$ metrics, using the outcome of 15 independent experiments. The variance between different trials is smaller than 0.001 for all evaluation metrics.

[3] https://github.com/SmartDataAnalytics/AK-DE-biGRU.

[4] Command descriptions scraped from Ubuntu man pages.

Model	R$_{10}$@1	R$_{10}$@3	R$_{10}$@5
DE-GRU	0.685	0.831	0.960
DE-biGRU	0.678	0.813	0.956
A-DE-GRU	0.712	0.845	0.964
A-DE-biGRU	0.739	0.864	0.968
AK$_+$-DE-biGRU	0.743	0.867	0.969
AK-DE-biGRU$_{w2v}$	0.745	0.866	0.970
AK-DE-biGRU	**0.747**	**0.868**	**0.972**

Table 3: Ablation study with different settings.

6.2 Ablation study

Our model differs in various ways from the vanilla DE: it uses a GRU instead of an LSTM for the encoding, introduces an attention mechanism for the encoding of the context and another for the encoding of the response, and incorporates additional knowledge in the response encoding process.

To analyze the effect of these components on the over all performance, we analyzed different model variants: a DE using a GRU or a bidirectional GRU as encoder (**DE-GRU** and **DE-biGRU**, respectively) and both of these models with attention augmented encoding for embedding both context and response (**A-DE-GRU** and **A-DE-biGRU**, respectively). We also tested the effects of using a simple addition instead of the weighted summation given in equation (10) for merging the word embedding with the desciption embedding (**AK$_+$-DE-biGRU**). Finally, we investigated a version of our model (**AK-DE-biGRU$_{w2v}$**) where we used pre-trained word2vec embeddings, as done by Wu et al. (2017), instead of learning our own word embeddings from the data set.

The results of the study are presented in Table 3. With the basic models, i.e. DE-GRU and DE-biGRU, as baselines, we observed around 4% and 9% improvement on R_{10}@1 when incorporating the attention mechanism (A-DE-GRU and A-DE-biGRU, respectively).

When domain knowledge is incorporated by simple addition (as in the work of Bahdanau et al. (2018)), i.e. in AK$_+$-DE-biGRU, we noticed 0.5% further improvement. Note however, that the results are not as good as when using the proposed weighted addition. Finally, using our method of incorporating domain knowledge in combination with embeddings trained from scratch with fastText (Bojanowski et al., 2016), the performance gets 0.3% better than when using pre-trained word2vec embeddings. In total, compared to the DE-biGRU baseline, our model (AK-DE-biGRU) achieves 10% of improvement in terms of the R_{10}@1 metric. Thus, the results clearly suggest that both the attention mechanism and the incorporation of domain knowledge, are effective approaches for improving the dual encoder architecture. Curiously, we noticed that for the baseline models, using a GRU as the encoder is better than using a biGRU. This finding is in line with the results from Kadlec et al. (2015) reported in Table 2. However, the table is turned when augmenting the models with an attention mechanism where the biGRU-based model outperforms the one with the GRU. This observation motivates us to consider a biGRU instead of a GRU in our final model.

Example Response Utterances
gui for shutdown try typing *sudo* *shutdown* -h now
sudo *apt-get* install qt4-designer there could be some qt dev packages too but i think the above will install them as dependencies
certainly won n't make a difference i m sure but maybe try *sudo* *shutdown* -r now shutdown works just fine graphical and command line
pci can you put the output of *lspci* on __url__ and give me the link please
i do n't see a line in xorg conf for hsync and vsync do you get the same you d create it i m looking at gentoo and ubuntu forums a sec
can be many reasons of *traceroute* __url__ you will not get a complete result

Table 4: Visualization of attention weight in utterance samples, darker shade means higher attention weight.

6.3 Visualizing response attentions

To further investigate the results given by our model, we qualitatively inspected several samples of response utterances and their attention weights, as shown in Table 4. We noticed that our model learned to focus on technical terms, such as *lspci*,

Context utterances
Utterance 1: Ubuntu <version>
Utterance 2: hi all sony vaio fx120 will not turn off when shutting down, any ideas? btw acpi =o ff in boot parameters anything else i should be trying?
Utterance 3: how are you shutting down i.e. terminal or gui?

Table 5: Sample context utterances from UDC's test set whose correct response is the first utterance in Table 4.

shutdown, and *traceroute*. We also observed that the model is able to capture contextual importance, i.e. it is able to focus on context relevant words. For example, given the context in Table 5 and the correct response in the first row of Table 4, one can see the attention on the word *shutdown*, where it gets a lower weight when used as a common word in the first occurance than as a UNIX command in the second. [5]

6.4 Error analysis

We qualitatively analyzed the errors our method made. We observed that our model's predictions are biased toward high information utterances. That is, we observed for some examples that the correct response is generic (i.e. has low information), our model chooses a non-generic response, as shown in Table 6. Furthermore, we computed the average utterance information content (the entropy) for both the correct and predicted responses, based on Xu and Reitter (2018), where we obtained 9.25 bits and 9.34 bits, respectively. This quantitatively indicates that our model is slightly biased toward high information responses.

7 Conclusion and future work

We presented a novel model which extends the dual encoder architecture for multi-turn response selection by incorporating external domain knowledge and attention augmented encoding. Our experimental results demonstrate that our model outperformed other state-of-the-art methods for response selection in a multi-turn dialogue setting, and that the attention mechanism and incorporating additional domain knowledge are indeed effective approaches for improving the response selection performance of the dual encoder architecture. Further improvement might be made by also considering domain knowledge in the context and by improving the handling of OOV words, e.g. by widening our domain specific word vocabulary and handling generic OOV words such as typos.

Examples of model error:
Correct: ok will do :), nope. **Predicted**: __url__ if you go down to the bottom of that tutorial i also have a post there that is a bit more detailed about my problem poster name is trent
Correct: hmm! ok **Predicted**: as did i w/ fbsd ... just check out the livecd for a bit
Correct: okay thank you a thread i hope :) **Predicted**: hmm ok because im not sure about iwconfig and wpa but we can give it a try do gksudo gedit __path__ then add a record like this __url__
Correct: right .. it is, it exists i verified **Predicted**: i want to connect to your computer remotely if you allow me to so i can fix the problem for you just follow the following procedure.
Correct: roger .. lemme check, got it ... thanks dude :) **Predicted**: just click the partition and then click the blue text next to mount point or you can simply navigate to that path

Table 6: Examples on the error our model made. We observed that our model's predictions are biased towards non-generic responses.

[5] N.B. The conversations are taken directly from the corpus and can be grammatically inconsistent.

References

Guozhen An, Mehrnoosh Shafiee, and Davood Shamsi. 2018. Improving retrieval modeling using cross convolution networks and multi frequency word embedding. *arXiv preprint arXiv:1802.05373*.

Dzmitry Bahdanau, Tom Bosc, Stanisław Jastrzebski, Edward Grefenstette, Pascal Vincent, and Yoshua Bengio. 2018. Learning to compute word embeddings on the fly.

Leemon Baird. 1995. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pages 30–37. Elsevier.

Petr Baudiš, Jan Pichl, Tomáš Vyskočil, and Jan Šedivỳ. 2016. Sentence pair scoring: Towards unified framework for text comprehension. *arXiv preprint arXiv:1603.06127*.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Antoine Bordes, Y-Lan Boureau, and Jason Weston. 2017. Learning end-to-end goal-oriented dialog. In *Proceedings of the 3rd International Conference for Learning Representations*.

Kyunghyun Cho, B van Merrienboer, Caglar Gulcehre, F Bougares, H Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*.

Jianxiong Dong and Jim Huang. 2018. Enhance word representation for out-of-vocabulary on ubuntu dialogue corpus. *arXiv preprint arXiv:1802.02614*.

Wang Hao, Lu Zhengdong, Li Hang, et al. 2013. A dataset for research on short-text conversation. In *Proceed-ings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 935–945.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Zongcheng Ji, Zhengdong Lu, and Hang Li. 2014. An information retrieval approach to short text conversation. In *Proceedings of International Conference on Computation and Language*.

Rudolf Kadlec, Martin Schmid, and Jan Kleindienst. 2015. Improved deep learning baselines for ubuntu corpus dialogs. In *NIPS on Machine Learning for Spoken Language Understanding*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference for Learning Representations*.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015a. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of SIGDIAL*.

Ryan Lowe, Nissan Pow, IV Serban, Laurent Charlin, and Joelle Pineau. 2015b. Incorporating unstructured textual knowledge sources into neural dialogue systems. In *Neural Information Processing Systems Workshop on Machine Learning for Spoken Language Understanding*.

Minh-Thang Luong, Ilya Sutskever, Quoc V Le, Oriol Vinyals, and Wojciech Zaremba. 2015. Addressing the rare word problem in neural machine translation. In *Proceeding of Association for Computational Linguistics*.

Johanna Moore Mary Ellen Foster Oliver and Lemon Michael White. 2004. Generating tailored, comparative descriptions in spoken dialogue.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Alan Ritter, Colin Cherry, and William B Dolan. 2011. Data-driven response generation in social media. In *Proceedings of the conference on empirical methods in natural language processing*, pages 583–593. Association for Computational Linguistics.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, volume 16, pages 3776–3784.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Proceedings of 53rd Annual Meeting of the Association for Computational Linguistics*.

Amanda Stent, Marilyn A Walker, Steve Whittaker, and Preetam Maloor. 2002. User-tailored generation for spoken dialogue: An experiment. In *Seventh International Conference on Spoken Language Processing*.

Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2016. Lstm-based deep learning models for non-factoid answer selection. In *Proceedings of the 4rd International Conference for Learning Representations*.

David C Uthus and David W Aha. 2013. Extending word highlighting in multiparticipant chat. In *FLAIRS Conference*.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. In *International Conference on Machine Learning: Deep Learning Workshop*.

Marilyn A Walker, Rebecca Passonneau, and Julie E Boland. 2001. Quantitative and qualitative evaluation of darpa communicator spoken dialogue systems. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 515–522. Association for Computational Linguistics.

Shengxian Wan, Yanyan Lan, Jun Xu, Jiafeng Guo, Liang Pang, and Xueqi Cheng. 2016. Match-srnn: Modeling the recursive matching structure with spatial rnn. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*.

Shuohang Wang and Jing Jiang. 2016. Learning natural language inference with lstm. In *Proceedings of NAACL-HLT 2016*.

Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *European Chapter of the Association for Computational Linguistics*.

Jason Williams, Antoine Raux, Deepak Ramachandran, and Alan Black. 2013. The dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*, pages 404–413.

Yu Wu, Wei Wu, Zhoujun Li, and Ming Zhou. 2016. Topic augmented neural network for short text conversation. *CoRR abs/1605.00090*.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 496–505.

Yang Xu and David Reitter. 2018. Information density converges in dialogue: Towards an information-theoretic model. *Cognition*, 170:147–163.

Zhen Xu, Bingquan Liu, Baoxun Wang, Chengjie Sun, and Xiaolong Wang. 2016. Incorporating loose-structured knowledge into lstm with recall gate for conversation modeling. *arXiv preprint arXiv:1605.05110*.

Rui Yan, Yiping Song, and Hua Wu. 2016. Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 55–64. ACM.

Tiancheng Zhao and Maxine Eskenazi. 2016. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. In *Proceedings of SIGDIAL*.

Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Dianhai Yu, Hao Tian, Xuan Liu, and Rui Yan. 2016. Multi-view response selection for human-computer conversation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 372–381.