Multi-Turn Response Selection for Chatbots with Deep Attention Matching Network

Xiangyang Zhou[*], **Lu Li**[*], **Daxiang Dong, Yi Liu, Ying Chen,**
Wayne Xin Zhao[†], **Dianhai Yu** and **Hua Wu**
Baidu Inc., Beijing, China
{zhouxiangyang, lilu12, dongdaxiang, liuyi05,
chenying04, v_zhaoxin, yudianhai, wu_hua}@baidu.com

Abstract

Human generates responses relying on semantic and functional dependencies, including coreference relation, among dialogue elements and their context. In this paper, we investigate matching a response with its multi-turn context using dependency information based entirely on attention. Our solution is inspired by the recently proposed Transformer in machine translation (Vaswani et al., 2017) and we extend the attention mechanism in two ways. First, we construct representations of text segments at different granularities solely with stacked self-attention. Second, we try to extract the truly matched segment pairs with attention across the context and response. We jointly introduce those two kinds of attention in one uniform neural network. Experiments on two large-scale multi-turn response selection tasks show that our proposed model significantly outperforms the state-of-the-art models.

1 Introduction

Building a chatbot that can naturally and consistently converse with human-beings on open-domain topics draws increasing research interests in past years. One important task in chatbots is *response selection*, which aims to select the best-matched response from a set of candidates given the context of a conversation. Besides playing a critical role in *retrieval-based chatbots* (Ji et al., 2014), response selection models have been used in *automatic evaluation of dialogue generation* (Lowe et al., 2017) and the discriminator of *GAN*-based (Generative Adversarial Networks) neural dialogue generation (Li et al., 2017).

Conversation Context

Speaker A: Hi I am looking to see what packages are installed on my system, I don't see a path, is the list being held somewhere else?
Speaker B: Try dpkg - get-selections
Speaker A: What is that like? A database for packages instead of a flat file structure?
Speaker B: dpkg is the debian package manager - get-selections simply shows you what packages are handed by it

Response of Speaker A: No clue what do you need it for, its just reassurance as I don't know the debian package manager

Figure 1: Example of human conversation on Ubuntu system troubleshooting. Speaker A is seeking for a solution of package management in his/her system and speaker B recommend using, the debian package manager, dpkg. But speaker A does not know dpkg, and asks a *backchannel-question* (Stolcke et al., 2000), i.e., "no clue what do you need it for?", aiming to double-check if dpkg could solve his/her problem. Text segments with underlines in the same color across context and response can be seen as matched pairs.

Early studies on response selection only use the last utterance in context for matching a reply, which is referred to as *single-turn response selection* (Wang et al., 2013). Recent works show that the consideration of a multi-turn context can facilitate selecting the next utterance (Zhou et al., 2016; Wu et al., 2017). The reason why richer contextual information works is that human generated responses are heavily dependent on the previous dialogue segments at different granularities (words, phrases, sentences, etc), both semantically and functionally, over multiple turns rather than one turn (Lee et al., 2006; Traum and Heeman, 1996). Figure 1 illustrates semantic connectivities between segment pairs across context and response. As demonstrated, generally there are two kinds of matched segment pairs at different granularities across context and response: (1) surface text relevance, for example the lexical overlap of words "packages"-"package" and phrases "debian package manager"-"debian pack-

[*] Equally contributed.
[†] Work done as a visiting scholar at Baidu. Wayne Xin Zhao is an associate professor of Renmin University of China and can be reached at batmanfly@ruc.edu.cn.

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers), pages 1118–1127
Melbourne, Australia, July 15 - 20, 2018.

age manager". (2) latent dependencies upon which segments are semantically/functionally related to each other. Such as the word "it" in the response, which refers to "dpkg" in the context, as well as the phrase "its just reassurance" in the response, which latently points to "what packages are installed on my system", the question that speaker A wants to double-check.

Previous studies show that capturing those matched segment pairs at different granularities across context and response is the key to multi-turn response selection (Wu et al., 2017). However, existing models only consider the textual relevance, which suffers from matching response that latently depends on previous turns. Moreover, Recurrent Neural Networks (RNN) are conveniently used for encoding texts, which is too costly to use for capturing multi-grained semantic representations (Lowe et al., 2015; Zhou et al., 2016; Wu et al., 2017). As an alternative, we propose to match a response with multi-turn context using dependency information based entirely on attention mechanism. Our solution is inspired by the recently proposed Transformer in machine translation (Vaswani et al., 2017), which addresses the issue of sequence-to-sequence generation only using attention, and we extend the key attention mechanism of Transformer in two ways:

self-attention By making a sentence attend to itself, we can capture its intra word-level dependencies. Phrases, such as "debian package manager", can be modeled with word-level self-attention over word-embeddings, and sentence-level representations can be constructed in a similar way with phrase-level self-attention. By hierarchically stacking self-attention from word embeddings, we can gradually construct semantic representations at different granularities.

cross-attention By making context and response attend to each other, we can generally capture dependencies between those latently matched segment pairs, which is able to provide complementary information to textual relevance for matching response with multi-turn context.

We jointly introduce self-attention and cross-attention in one uniform neural matching network, namely the Deep Attention Matching Network (DAM), for multi-turn response selection. In practice, DAM takes each single word of an utterance in context or response as the centric-meaning of an abstractive semantic segment, and hierarchically enriches its representation with stacked self-attention, gradually producing more and more sophisticated segment representations surrounding the centric-word. Each utterance in context and response are matched based on segment pairs at different granularities, considering both textual relevance and dependency information. In this way, DAM generally captures matching information between the context and the response from word-level to sentence-level, important matching features are then distilled with convolution & max-pooling operations, and finally fused into one single matching score via a single-layer perceptron.

We test DAM on two large-scale public multi-turn response selection datasets, the Ubuntu Corpus v1 and Douban Conversation Corpus. Experimental results show that our model significantly outperforms the state-of-the-art models, and the improvement to the best baseline model on $\mathbf{R}_{10}@1$ is over 4%. What is more, DAM is expected to be convenient to deploy in practice because most attention computation can be fully parallelized (Vaswani et al., 2017). Our contributions are two-folds: (1) we propose a new matching model for multi-turn response selection with self-attention and cross-attention. (2) empirical results show that our proposed model significantly outperforms the state-of-the-art baselines on public datasets, demonstrating the effectiveness of self-attention and cross-attention.

2 Related Work

2.1 Conversational System

To build an automatic conversational agent is a long cherished goal in Artificial Intelligence (AI) (Turing, 1950). Previous researches include *task-oriented dialogue system*, which focuses on completing tasks in vertical domain, and *chatbots*, which aims to consistently and naturally converse with human-beings on open-domain topics. Most modern chatbots are data-driven, either in a fashion of information-retrieval (Ji et al., 2014; Banchs and Li, 2012; Nio et al., 2014; Ameixa et al., 2014) or sequence-generation (Ritter et al., 2011). The retrieval-based systems enjoy the advantage of informative and fluent responses because it searches a large dialogue repository and selects

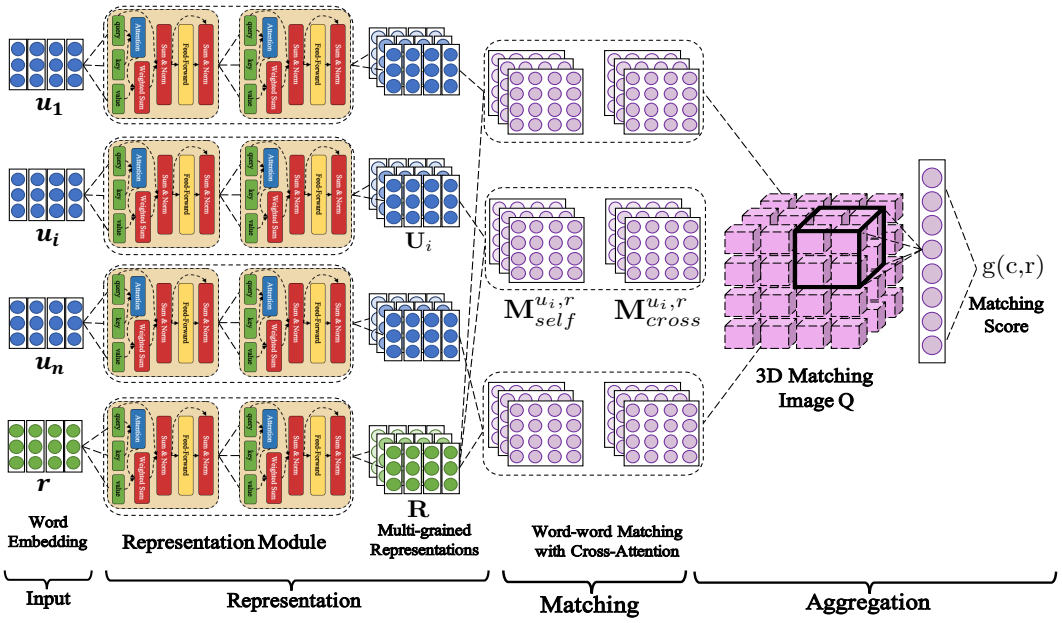

Figure 2: Overview of Deep Attention Matching Network.

candidate that best matches the current context. The generation-based models, on the other hand, learn patterns of responding from dialogues and can directly generalize new responses.

2.2 Response Selection

Researches on response selection can be generally categorized into *single-turn* and *multi-turn*. Most early studies are single-turn that only consider the last utterance for matching response (Wang et al., 2013, 2015). Recent works extend it to multi-turn conversation scenario, Lowe et al.,(2015) and Zhou et al.,(2016) use RNN to read context and response, use the last hidden states to represent context and response as two semantic vectors, and measure their relevance. Instead of only considering the last states of RNN, Wu et al.,(2017) take hidden state at each time step as a text segment representation, and measure the distance between context and response via segment-segment matching matrixes. Nevertheless, matching with dependency information is generally ignored in previous works.

2.3 Attention

Attention has been proven to be very effective in Natural Language Processing (NLP) (Bahdanau et al., 2015; Yin et al., 2016; Lin et al., 2017) and other research areas (Xu et al., 2015). Recently, Vaswani et al.,(2017) propose a novel sequence-to-sequence generation network, the Transformer, which is entirely based on attention. Not only Transformer can achieve better translation results than convenient RNN-based models, but also it is very fast in training/predicting as the computation of attention can be fully parallelized. Previous works on attention mechanism show the superior ability of attention to capture semantic dependencies, which inspires us to improve multi-turn response selection with attention mechanism.

3 Deep Attention Matching Network

3.1 Problem Formalization

Given a dialogue data set $\mathcal{D} = \{(c, r, y)_Z\}_{Z=1}^{N}$, where $c = \{u_0, ..., u_{n-1}\}$ represents a conversation context with $\{u_i\}_{i=0}^{n-1}$ as utterances and r as a response candidate. $y \in \{0, 1\}$ is a binary label, indicating whether r is a proper response for c. Our goal is to learn a matching model $g(c, r)$ with $\mathcal{D}$, which can measure the relevance between any context c and candidate response r.

3.2 Model Overview

Figure 2 gives an overview of DAM, which generally follows the **representation-matching-aggregation** framework to match response with multi-turn context. For each utterance $u_i = [w_{u_i,k}]_{k=0}^{n_{u_i}-1}$ in a context and its response candidate $r = [w_{r,t}]_{t=0}^{n_r-1}$, where n_{u_i} and n_r stand for the numbers of words, DAM first looks up a shared word embedding table and represents u_i and r as sequences of word embeddings, namely $\mathbf{U}_i^0 =$

$[e^0_{u_i,0}, ..., e^0_{u_i,n_{u_i}-1}]$ and $\mathbf{R}^0 = [e^0_{r,0}, ..., e^0_{r,n_r-1}]$ respectively, where $e \in \mathbb{R}^d$ denotes a d-dimension word embedding.

A representation module then starts to construct semantic representations at different granularities for u_i and r. Practically, L identical layers of self-attention are hierarchically stacked, each l^{th} self-attention layer takes the output of the $l-1^{th}$ layer as its input, and composites the input semantic vectors into more sophisticated representations based on self-attention. In this way, multi-grained representations of u_i and r are gradually constructed, denoted as $[\mathbf{U}_i^l]_{l=0}^L$ and $[\mathbf{R}^l]_{l=0}^L$ respectively.

Given $[\mathbf{U}_i^0, ..., \mathbf{U}_i^L]$ and $[\mathbf{R}^0, ..., \mathbf{R}^L]$, utterance u_i and response r are then matched with each other in a manner of segment-segment similarity matrix. Practically, for each granularity $l \in [0...L]$, two kinds of matching matrixes are constructed, i.e., the self-attention-match $\mathbf{M}_{self}^{u_i,r,l}$ and cross-attention-match $\mathbf{M}_{cross}^{u_i,r,l}$, measuring the relevance between utterance and response with textual information and dependency information respectively.

Those matching scores are finally merged into a 3D matching image $\mathbf{Q}$[1]. Each dimension of $\mathbf{Q}$ represents *each utterance in context*, *each word in utterance* and *each word in response* respectively. Important matching information between segment pairs across multi-turn context and candidate response is then extracted via convolution with max-pooling operations, and further fused into one matching score via a single-layer perceptron, representing the matching degree between the response candidate and the whole context.

Specifically, we use a shared component, the Attentive Module, to implement both self-attention in representation and cross-attention in matching. We will discuss in detail the implementation of Attentive Module and how we used it to implement both self-attention and cross-attention in following sections.

3.3 Attentive Module

Figure 3 shows the structure of Attentive Module, which is similar to that used in Transformer (Vaswani et al., 2017). Attentive Module has three input sentences: the query sentence, the key sentence and the value sentence, namely $\mathcal{Q} = [e_i]_{i=0}^{n_\mathcal{Q}-1}, \mathcal{K} = [e_i]_{i=0}^{n_\mathcal{K}-1}, \mathcal{V} = [e_i]_{i=0}^{n_\mathcal{V}-1}$ respectively, where $n_\mathcal{Q}$, $n_\mathcal{K}$ and $n_\mathcal{V}$ denote the number of words in each sentence and e_i stands for a d-dimension embedding, $n_\mathcal{K}$ is equal to $n_\mathcal{V}$. The Attentive Module first takes each word in the query sentence to attend to words in the key sentence via Scaled Dot-Product Attention (Vaswani et al., 2017), then applies those attention results upon the value sentence, which is defined as:

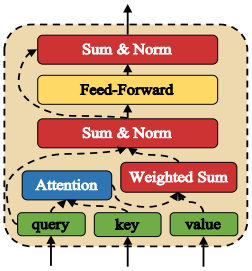

Figure 3: Attentive Module.

$$Att(\mathcal{Q}, \mathcal{K}) = \left[softmax(\frac{\mathcal{Q}[i] \cdot \mathcal{K}^T}{\sqrt{d}})\right]_{i=0}^{n_\mathcal{Q}-1} \quad (1)$$

$$\mathcal{V}_{att} = Att(\mathcal{Q}, \mathcal{K}) \cdot \mathcal{V} \in \mathbb{R}^{n_\mathcal{Q} \times d} \quad (2)$$

where $\mathcal{Q}[i]$ is the i^{th} embedding in the query sentence $\mathcal{Q}$. Each row of $\mathcal{V}_{att}$, denoted as $\mathcal{V}_{att}[i]$, stores the fused semantic information of words in the value sentence that possibly have dependencies to the i^{th} word in query sentence. For each i, $\mathcal{V}_{att}[i]$ and $\mathcal{Q}[i]$ are then added up together, compositing them into a new representation that contains their joint meanings. A layer normalization operation (Ba et al., 2016) is then applied, which prevents vanishing or exploding of gradients. A feed-forward network **FFN** with RELU (LeCun et al., 2015) activation is then applied upon the normalization result, in order to further process the fused embeddings, defined as:

$$\mathbf{FFN}(x) = max(0, xW_1 + b_1)W_2 + b_2 \quad (3)$$

where, x is a 2D-tensor in the same shape of query sentence $\mathcal{Q}$ and W_1, b_1, W_2, b_2 are learnt parameters. This kind of activation is empirically useful in other works, and we also adapt it in our model. The result $\mathbf{FFN}(x)$ is a 2D-tensor that has the same shape as x, $\mathbf{FFN}(x)$ is then residually added (He et al., 2016) to x, and the fusion result is then normalized as the final outputs. We refer to the whole Attentive Module as:

$$\mathbf{AttentiveModule}(\mathcal{Q}, \mathcal{K}, \mathcal{V}) \quad (4)$$

[1]We refer to it as **Q** because it is like a cube.

As described, Attentive Module can capture dependencies across query sentence and key sentence, and further use the dependency information to composite elements in the query sentence and the value sentence into compositional representations. We exploit this property of the Attentive Module to construct multi-grained semantic representations as well as match with dependency information.

3.4 Representation

Given $\mathbf{U}_i^0$ or $\mathbf{R}^0$, the word-level embedding representations for utterance u_i or response r, DAM takes $\mathbf{U}_i^0$ ro $\mathbf{R}^0$ as inputs and hierarchically stacks the Attentive Module to construct multi-grained representations of u_i and r, which is formulated as:

$$\mathbf{U}_i^{l+1} = \textbf{AttentiveModule}(\mathbf{U}_i^l, \mathbf{U}_i^l, \mathbf{U}_i^l) \quad (5)$$

$$\mathbf{R}^{l+1} = \textbf{AttentiveModule}(\mathbf{R}^l, \mathbf{R}^l, \mathbf{R}^l) \quad (6)$$

where l ranges from 0 to $L-1$, denoting the different levels of granularity. By this means, words in each utterance or response repeatedly function together to composite more and more holistic representations, we refer to those multi-grained representations as $[\mathbf{U}_i^0, ..., \mathbf{U}_i^L]$ and $[\mathbf{R}^0, ..., \mathbf{R}^L]$ hereafter.

3.5 Utterance-Response Matching

Given $[\mathbf{U}_i^l]_{l=0}^L$ and $[\mathbf{R}^l]_{l=0}^L$, two kinds of segment-segment matching matrixes are constructed at each level of granularity l, i.e., the self-attention-match $\mathbf{M}_{self}^{u_i,r,l}$ and cross-attention-match $\mathbf{M}_{cross}^{u_i,r,l}$. $\mathbf{M}_{self}^{u_i,r,l}$ is defined as:

$$\mathbf{M}_{self}^{u_i,r,l} = \{\mathbf{U}_i^l[k]^T \cdot \mathbf{R}^l[t]\}_{n_{u_i} \times n_r} \quad (7)$$

in which, each element in the matrix is the dot-product of $\mathbf{U}_i^l[k]$ and $\mathbf{R}^l[t]$, the k^{th} embedding in $\mathbf{U}_i^l$ and the t^{th} embedding in $\mathbf{R}^l$, reflecting the textual relevance between the k^{th} segment in u_i and t^{th} segment in r at the l^{th} granularity. The cross-attention-match matrix is based on cross-attention, which is defined as:

$$\widetilde{\mathbf{U}}_i^l = \textbf{AttentiveModule}(\mathbf{U}_i^l, \mathbf{R}^l, \mathbf{R}^l) \quad (8)$$

$$\widetilde{\mathbf{R}}^l = \textbf{AttentiveModule}(\mathbf{R}^l, \mathbf{U}_i^l, \mathbf{U}_i^l) \quad (9)$$

$$\mathbf{M}_{cross}^{u_i,r,l} = \{\widetilde{\mathbf{U}}_i^l[k]^T \cdot \widetilde{\mathbf{R}}^l[t]\}_{n_{u_i} \times n_r} \quad (10)$$

where we use Attentive Module to make $\mathbf{U}_i^l$ and $\mathbf{R}^l$ crossly attend to each other, constructing two new representations for both of them, written as $\widetilde{\mathbf{U}}_i^l$ and $\widetilde{\mathbf{R}}^l$ respectively. Both $\widetilde{\mathbf{U}}_i^l$ and $\widetilde{\mathbf{R}}^l$ implicitly capture semantic structures that cross the utterance and response. In this way, those inter-dependent segment pairs are close to each other in representations, and dot-products between those latently inter-dependent pairs could get increased, providing dependency-aware matching information.

3.6 Aggregation

DAM finally aggregates all the segmental matching degrees across each utterance and response into a 3D matching image $\mathbf{Q}$, which is defined as:

$$\mathbf{Q} = \{\mathbb{Q}_{i,k,t}\}_{n \times n_{u_i} \times n_r} \quad (11)$$

where each pixel $\mathbb{Q}_{i,k,t}$ is formulated as:

$$\mathbb{Q}_{i,k,t} = [\mathbf{M}_{self}^{u_i,r,l}[k,t]]_{l=0}^L \oplus [\mathbf{M}_{cross}^{u_i,r,l}[k,t]]_{l=0}^L \quad (12)$$

$\oplus$ is concatenation operation, and each pixel has $2(L+1)$ channels, storing the matching degrees between one certain segment pair at different levels of granularity. DAM then leverages two-layered 3D convolution with max-pooling operations to distill important matching features from the whole image. The operation of 3D convolution with max-pooling is the extension of typical 2D convolution, whose filters and strides are 3D cubes[2]. We finally compute matching score $g(c,r)$ based on the extracted matching features $f_{match}(c,r)$ via a single-layer perceptron, which is formulated as:

$$g(c,r) = \sigma(W_3 f_{match}(c,r) + b_3) \quad (13)$$

where W_3 and b_3 are learnt parameters, and σ is sigmoid function that gives the probability if r is a proper candidate to c. The loss function of DAM is the negative log likelihood, defined as:

$$p(y|c,r) = g(c,r)y + (1 - g(c,r))(1-y) \quad (14)$$

$$L(\cdot) = -\sum_{(c,r,y) \in \mathcal{D}} log(p(y|c,r)) \quad (15)$$

4 Experiment

[2] https://www.tensorflow.org/api_docs/python/tf/nn/conv3d

	Ubuntu Corpus				Douban Conversation Corpus					
	$\mathbf{R}_2$@1	$\mathbf{R}_{10}$@1	$\mathbf{R}_{10}$@2	$\mathbf{R}_{10}$@5	MAP	MRR	P@1	$\mathbf{R}_{10}$@1	$\mathbf{R}_{10}$@2	$\mathbf{R}_{10}$@5
DualEncoder$_{lstm}$	0.901	0.638	0.784	0.949	0.485	0.527	0.320	0.187	0.343	0.720
DualEncoder$_{bilstm}$	0.895	0.630	0.780	0.944	0.479	0.514	0.313	0.184	0.330	0.716
MV-LSTM	0.906	0.653	0.804	0.946	0.498	0.538	0.348	0.202	0.351	0.710
Match-LSTM	0.904	0.653	0.799	0.944	0.500	0.537	0.345	0.202	0.348	0.720
Multiview	0.908	0.662	0.801	0.951	0.505	0.543	0.342	0.202	0.350	0.729
DL2R	0.899	0.626	0.783	0.944	0.488	0.527	0.330	0.193	0.342	0.705
SMN$_{dynamic}$	0.926	0.726	0.847	0.961	0.529	0.569	0.397	0.233	0.396	0.724
DAM	**0.938**	**0.767**	**0.874**	**0.969**	**0.550**	**0.601**	**0.427**	**0.254**	**0.410**	**0.757**
DAM$_{first}$	0.927	0.736	0.854	0.962	0.528	0.579	0.400	0.229	0.396	0.741
DAM$_{last}$	0.932	0.752	0.861	0.965	0.539	0.583	0.408	0.242	0.407	0.748
DAM$_{self}$	0.931	0.741	0.859	0.964	0.527	0.574	0.382	0.221	0.403	0.750
DAM$_{cross}$	0.932	0.749	0.863	0.966	0.535	0.585	0.400	0.234	0.411	0.733

Table 1: Experimental results of DAM and other comparison approaches on Ubuntu Corpus V1 and Douban Conversation Corpus.

4.1 Dataset

We test DAM on two public multi-turn response selection datasets, the Ubuntu Corpus V1 (Lowe et al., 2015) and the Douban Conversation Corpus (Wu et al., 2017). The former one contains multi-turn dialogues about Ubuntu system troubleshooting in English and the later one is crawled from a Chinese social networking on open-domain topics. The Ubuntu training set contains 0.5 million multi-turn contexts, and each context has one positive response that generated by human and one negative response which is randomly sampled. Both validation and testing sets of Ubuntu Corpus have 50k contexts, where each context is provided with one positive response and nine negative replies. The Douban corpus is constructed in a similar way to the Ubuntu Corpus, except that its validation set contains 50k instances with 1:1 positive-negative ratios and the testing set of Douban corpus is consisted of 10k instances, where each context has 10 candidate responses, collected via a tiny inverted-index system (Lucene[3]), and labels are manually annotated.

4.2 Evaluation Metric

We use the same evaluation metrics as in previous works (Wu et al., 2017). Each comparison model is asked to select k best-matched response from n available candidates for the given conversation context c, and we calculate the recall of the true positive replies among the k selected ones as the main evaluation metric, denoted as $\mathbf{R}_n@k = \frac{\sum_{i=1}^{k} y_i}{\sum_{i=1}^{n} y_i}$, where y_i is the binary label for each candidate. In addition to $\mathbf{R}_n@k$, we use MAP (Mean Average Precision) (Baeza-Yates et al., 1999), MRR (Mean Reciprocal Rank) (Voorhees et al., 1999), and Precision-at-one $P@1$ especially for Douban corpus, following the setting of previous works (Wu et al., 2017).

4.3 Comparison Methods

RNN-based models : Previous best performing models are based on RNNs, we choose representative models as baselines, including SMN$_{dynamic}$(Wu et al., 2017), Multi-view(Zhou et al., 2016), DualEncoder$_{lstm}$ and DualEncoder$_{bilstm}$ (Lowe et al., 2015), DL2R (Yan et al., 2016), Match-LSTM (Wang and Jiang, 2017) and MV-LSTM (Pang et al., 2016), where SMN$_{dynamic}$ achieves the best scores against all the other published works, and we take it as our state-of-the-art baseline.

Ablation : To verify the effects of multi-grained representation, we setup two comparison models, i.e., DAM$_{first}$ and DAM$_{last}$, which dispense with the multi-grained representations in DAM, and use representation results from the 0^{th} layer and L^{th} layer of self-attention instead. Moreover, we setup DAM$_{self}$ and DAM$_{cross}$, which only use self-attention-match or cross-attention-match respectively, in order to examine the effectiveness of both self-attention-match and cross-attention-match.

4.4 Model Training

We copy the reported evaluation results of all baselines for comparison. DAM is implemented in tensorflow[4], and the used vocabularies, word em-

[3]https://lucenent.apache.org/

[4]https://www.tensorflow.org. Our code and data will be available at https://github.com/baidu/Dialogue/DAM

bedding sizes for Ubuntu corpus and Douban corpus are all set as same as the SMN (Wu et al., 2017). We consider at most 9 turns and 50 words for each utterance (response) in our experiments, word embeddings are pre-trained using training sets via word2vec (Mikolov et al., 2013), similar to previous works. We use zero-pad to handle the variable-sized input and parameters in **FFN** are set to 200, same as word-embedding size. We test stacking 1-7 self-attention layers, and reported our results with 5 stacks of self-attention because it gains the best scores on validation set. The 1st convolution layer has 32 [3,3,3] filters with [1,1,1] stride, and its max-pooling size is [3,3,3] with [3,3,3] stride. The 2nd convolution layer has 16 [3,3,3] filters with [1,1,1] stride, and its max-pooling size is also [3,3,3] with [3,3,3] stride. We tune DAM and the other ablation models with adam optimizer (Le et al., 2011) to minimize loss function defined in Eq 15. Learning rate is initialized as 1e-3 and gradually decreased during training, and the batch-size is 256. We use validation sets to select the best models and report their performances on test sets.

4.5 Experiment Result

Table 1 shows the evaluation results of DAM as well as all comparison models. As demonstrated, DAM significantly outperforms other competitors on both Ubuntu Corpus and Douban Conversation Corpus, including $\text{SMN}_{dynamic}$, which is the state-of-the-art baseline, demonstrating the superior power of attention mechanism in matching response with multi-turn context. Besides, both the performances of DAM_{first} and DAM_{self} decrease a lot compared with DAM, which shows the effectiveness of self-attention and cross-attention. Both DAM_{first} and DAM_{last} underperform DAM, which demonstrates the benefits of using multi-grained representations. Also the absence of self-attention-match brings down the precision, as shown in DAM_{cross}, exhibiting the necessity of jointly considering textual relevance and dependency information in response selection.

One notable point is that, while DAM_{first} is able to achieve close performance to $\text{SMN}_{dynamic}$, it is about 2.3 times faster than $\text{SMN}_{dynamic}$ in our implementation as it is very simple in computation. We believe that DAM_{first} is more suitable to the scenario that has limitations in computation time or memories but requires high precise, such as industry application or working as an component in other neural networks like GANs.

5 Analysis

We use the Ubuntu Corpus for analyzing how self-attention and cross-attention work in DAM from both quantity analysis as well as visualization.

5.1 Quantity Analysis

We first study how DAM performs in different utterance number of context. The left part in Figure 4 shows the changes of $\mathbf{R}_{10}@1$ on Ubuntu Corpus across contexts with different number of utterance. As demonstrated, while being good at matching response with long context that has more than 4 utterances, DAM can still stably deal with short context that only has 2 turns.

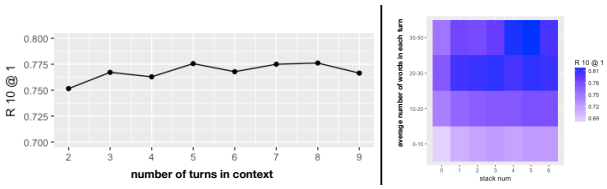

Figure 4: DAM's performance on Ubuntu Corpus across different contexts. The left part shows the performance in different utterance number of context. The right part shows performance in different average utterance text length of context as well as self-attention stack depth.

Moreover, the right part of Figure 4 gives the comparison of performance across different contexts with different average utterance text length and self-attention stack depth. As demonstrated, stacking self-attention can consistently improve matching performance for contexts having different average utterance text length, implying the stability advantage of using multi-grained semantic representations. The performance of matching short utterances, that have less than 10 words, is obviously lower than the other longer ones. This is because the shorter the utterance text is, the fewer information it contains, and the more difficult for selecting the next utterance, while stacking self-attention can still help in this case. However for long utterances like containing more than 30 words, stacking self-attention can significantly improve the matching performance, which means that the more information an utterance contains, the more stacked self-attention it needs to capture its intra semantic structures.

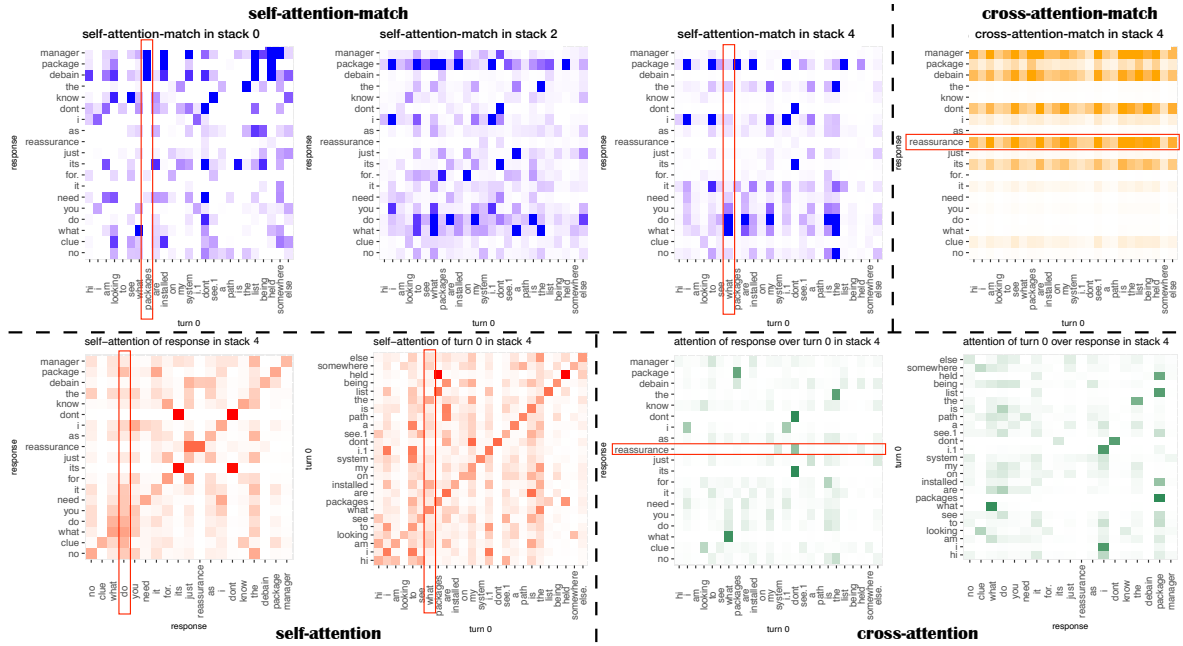

Figure 5: Visualization of self-attention-match, cross-attention-match as well as the distribution of self-attention and cross-attention in matching response with the first utterance in Figure 1. Each colored grid represents the matching degree or attention score between two words. The deeper the color is, the more important this grid is.

5.2 Visualization

We study the case in Figure 1 for analyzing in detail how self-attention and cross-attention work. Practically, we apply a softmax operation over self-attention-match and cross-attention-match, to examine the variance of dominating matching pairs during stacking self-attention or applying cross-attention. Figure 5 gives the visualization results of the 0^{th}, 2^{nd} and 4^{th} self-attention-match matrixes, the 4^{th} cross-attention-match matrix, as well as the distribution of self-attention and cross-attention in the 4^{th} layer in matching response with the first utterance (turn 0) due to space limitation. As demonstrated, important matching pairs in *self-attention-match in stack 0* are nouns, verbs, like "package" and "packages", those are similar in topics. However matching scores between prepositions or pronouns pairs, such as "do" and "what", become more important in *self-attention-match in stack 4*. The visualization results of self-attention show the reason why matching between prepositions or pronouns matters, as demonstrated, self-attention generally capture the semantic structure of "no clue what do you need package manager" for "do" in response and "what packages are installed" for "what" in utterance, making segments surrounding "do" and "what" close to each other in representations, thus increases their dot-product results.

Also as shown in Figure 5, self-attention-match and cross-attention-match capture complementary information in matching utterance with response. Words like "reassurance" and "its" in response significantly get larger matching scores in cross-attention-match compared with self-attention-match. According to the visualization of cross-attention, "reassurance" generally depends on "system" "don't" and "held" in utterance, which makes it close to words like "list", "installed" or "held" of utterance. Scores of cross-attention-match trend to centralize on several segments, which probably means that those segments in response generally capture structure-semantic information across utterance and response, amplifying their matching scores against the others.

5.3 Error Analysis

To understand the limitations of DAM and where the future improvements might lie, we analyze 100 strong bad cases from test-set that fail in $\mathbf{R}_{10}@5$. We find two major kinds of bad cases: (1) **fuzzy-candidate**, where response candidates are basically proper for the conversation context, except for a few improper details. (2) **logical-error**, where response candidates are wrong due to logical mismatch, for example, given a conversation context **A:** "I just want to stay at home tomorrow.", **B:** "Why not go hiking? I can go with

you.", response candidate like "Sure, I was planning to go out tomorrow." is logically wrong because it is contradictory to the first utterance of speaker A. We believe generating adversarial examples, rather than randomly sampling, during training procedure may be a good idea for addressing both **fuzzy-candidate** and **logical-error**, and to capture logic-level information hidden behind conversation text is also worthy to be studied in the future.

6 Conclusion

In this paper, we investigate matching a response with its multi-turn context using dependency information based entirely on attention. Our solution extends the attention mechanism of Transformer in two ways: (1) using stacked self-attention to harvest multi-grained semantic representations. (2) utilizing cross-attention to match with dependency information. Empirical results on two large-scale datasets demonstrate the effectiveness of self-attention and cross-attention in multi-turn response selection. We believe that both self-attention and cross-attention could benefit other research area, including spoken language understanding, dialogue state tracking or seq2seq dialogue generation. We would like to explore in depth how attention can help improve neural dialogue modeling for both chatbots and task-oriented dialogue systems in our future work.

Acknowledgement

We gratefully thank the anonymous reviewers for their insightful comments. This work is supported by the National Basic Research Program of China (973 program, No. 2014CB340505).

References

David Ameixa, Luisa Coheur, Pedro Fialho, and Paulo Quaresma. 2014. Luke, i am your father: dealing with out-of-domain requests by using movies subtitles. In *International Conference on Intelligent Virtual Agents*, pages 13–21. Springer.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. 1999. *Modern information retrieval*, volume 463. ACM press New York.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *international conference on learning representations*.

Rafael E Banchs and Haizhou Li. 2012. Iris: a chat-oriented dialogue system based on the vector space model. In *Proceedings of the ACL 2012 System Demonstrations*, pages 37–42. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Zongcheng Ji, Zhengdong Lu, and Hang Li. 2014. An information retrieval approach to short text conversation. *arXiv preprint arXiv:1408.6988*.

Quoc V Le, Jiquan Ngiam, Adam Coates, Abhik Lahiri, Bobby Prochnow, and Andrew Y Ng. 2011. On optimization methods for deep learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 265–272.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature*, 521(7553):436–444.

Alan Lee, Rashmi Prasad, Aravind Joshi, Nikhil Dinesh, and Bonnie Webber. 2006. Complexity of dependencies in discourse: Are dependencies in discourse more complex than in syntax. In *Proceedings of the 5th International Workshop on Treebanks and Linguistic Theories, Prague, Czech Republic*, page 12.

Jiwei Li, Will Monroe, Tianlin Shi, Alan Ritter, and Dan Jurafsky. 2017. Adversarial learning for neural dialogue generation. In *EMNLP*, pages 372–381.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *international conference on learning representations*.

Ryan Lowe, Michael Noseworthy, Iulian V Serban, Nicolas Angelard-Gontier, Yoshua Bengio, and Joelle Pineau. 2017. Towards an automatic turing test: Learning to evaluate dialogue responses. In *ACL*, pages 372–381.

Ryan Lowe, Nissan Pow, Iulian Vlad Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *annual meeting of the special interest group on discourse and dialogue*, pages 285–294.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Lasguido Nio, Sakriani Sakti, Graham Neubig, Tomoki Toda, Mirna Adriani, and Satoshi Nakamura. 2014. Developing non-goal dialog system based on examples of drama television. In *Natural Interaction with Robots, Knowbots and Smartphones*, pages 355–361. Springer.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. In *AAAI*, pages 2793–2799.

Alan Ritter, Colin Cherry, and William B Dolan. 2011. Data-driven response generation in social media. In *In Proc. EMNLP*, pages 583–593. Association for Computational Linguistics.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373.

David R Traum and Peter A Heeman. 1996. Utterance units in spoken dialogue. In *Workshop on Dialogue Processing in Spoken Language Systems*, pages 125–140.

Alan M Turing. 1950. Computing machinery and intelligence. *Mind*, 59(236):433–460.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Ellen M Voorhees et al. 1999. The trec-8 question answering track report. In *Trec*, pages 77–82.

Hao Wang, Zhengdong Lu, Hang Li, and Enhong Chen. 2013. A dataset for research on short-text conversations. In *EMNLP*, pages 935–945.

Mingxuan Wang, Zhengdong Lu, Hang Li, and Qun Liu. 2015. Syntax-based deep matching of short texts. *International Joint Conferences on Artificial Intelligence*.

Shuohang Wang and Jing Jiang. 2017. Machine comprehension using match-lstm and answer pointer. *international conference on learning representations*.

Yu Wu, Wei Wu, Ming Zhou, and Zhoujun Li. 2017. Sequential match network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *ACL*, pages 372–381.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057.

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.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association of Computational Linguistics*, 4(1):259–272.

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 *EMNLP*, pages 372–381.