Emulating Human Conversations using Convolutional Neural Network-based IR

Abhay Prakash
Microsoft, India
abprak@microsoft.com

Chris Brockett
Microsoft Research,
Redmond, WA, USA
chrisbkt@microsoft.com

Puneet Agrawal
Microsoft, India
punagr@microsoft.com

ABSTRACT

Conversational agents ("bots") are beginning to be widely used in conversational interfaces. To design a system that is capable of emulating human-like interactions, a conversational layer that can serve as a fabric for chat-like interaction with the agent is needed. In this paper, we introduce a model that employs Information Retrieval by utilizing convolutional deep structured semantic neural network-based features in the ranker to present human-like responses in ongoing conversation with a user. In conversations, accounting for context is critical to the retrieval model; we show that our context-sensitive approach using a Convolutional Deep Structured Semantic Model (cDSSM) with character trigrams significantly outperforms several conventional baselines in terms of the relevance of responses retrieved.

Categories and Subject Descriptors

H.3.3 [**Information Storage And Retrieval**]: Information Search and Retrieval; I.2.7 [**Artificial Intelligence**]: Natural Language Processing

Keywords

Chat bot; Deep learning; Structured Semantics; Conversational agent; Convolutional Networks; Twitter data

1. INTRODUCTION

Recent research in neural network-based Question Answering (QA), has focused primarily on providing the most relevant answer to a given query [7][9][10][11][18][19]. However, as businesses move towards building conversational interfaces (for example, Facebook M, Cortana, Siri, and Orat.ai) where humans and systems work collaboratively to achieve their goals, it becomes important to model the context surrounding the conversation, since the system is not only responding to a question, but needs to answer in the context of the history of the exchange.

Context is crucial, because conversational language does not always make for good query terms. Figure 1 presents an

User:	*June starts. My bday month. aww...*
Bot:	*for me too*
User:	*cool. yours on?*
Bot:	*mine is 25 June*

Figure 1: Example conversation between user and bot.

example of an interaction between a conversational agent ("bot") and user that resembles a dialog between friends. The bot responds, on the basis of the previous history of the conversation, *mine is 25 June*. It will be noted, however, that the user's messages in this conversation are casually colloquial and, taken in isolation, vaguely worded (*cool. yours on?*). Such queries are not easily addressable in traditional QA systems, which might need to perform complex anaphora resolution to determine that the query *yours on?* relates to the phrase *My bday month*.

In this paper, we present a deep neural network-based approach to implementing a conversational agent that will engage with users in a friendly, engaging, conversational fashion, drawing on a database of Twitter conversations. We model the task of providing a response as an Information Retrieval problem, i.e., for a given query issued by the user, the bot must retrieve the best candidate from a conversational corpus to output as response. Below, we use the following notation: we will refer to user message as M, agent response as R, and the previous history of conversation as C (context). We use features derived from a Convolutional Deep Structured Semantic Model (cDSSM) [12] to predict the output R given (M, C) on a corpus of Twitter conversations and demonstrate that use of cDSSM-based contextual features can boost the 1-best precision of retrieved responses over several alternatives.

2. RELATED WORK

A growing body of research has emerged in chat response generation, either using machine translation techniques [8] or deep neural networks, including sequence-to-sequence modeling [10][13][16]. These models, however, can be cumbersome to train, and may suffer a tendency towards blandness of output [5].

The system we explore here, by contrast, uses deep learning methods to retrieve the best response from a database of candidates. These candidates come from real human conversations and hence may be more vivid and less consensus-driven. In this respect, the system is close in spirit to the work of [15] on interactive storytelling where for a given mes-

Neu-IR'16 SIGIR Workshop on Neural Information Retrieval, July 21, 2016, Pisa, Italy

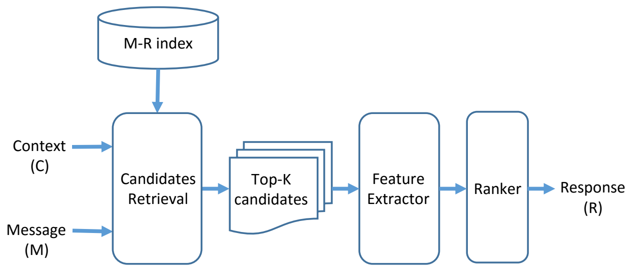

Figure 2: Our system is similar to a traditional IR system. We focus on extracting relevant features using deep learning.

sage (*user's sentence*) the system retrieves the best matching message (*a sentence in story corpus*) and returns with the corresponding response (*following sentence in story corpus*), using TF-IDF-based scoring.

The present work also bears a familial resemblance to [4] and the IR baseline in [13], where Twitter conversation pairs (each consisting of a tweet and its response) are used as data. Like our system, for a given query, those authors retrieve the best matching tweets to form a candidate set by taking their corresponding replies and rank them to display an appropriate response. [4] does not take into account context and does not deploy deep learning. [13] describe a neural language generation model (DCGM-II + CMM) that incorporates context to rank retrieval candidates. We use cDSSM, which is computationally efficient and relies on implicit semantic structure to derive features used in the ranker (described in Section 3.4).

Deep learning techniques have recently been applied in direct community question-answering in [7][9][19], and also in [17] who applied cDSSM. These approaches seek to return the answer from a knowledge base in triplet format. Similarly, [9] used a convolutional neural network approach to rank short text pairs when answering questions. Other recent neural retrieval models include a multilayer perceptron classifier [14] and three-layered neural networks [2][6] in the NTCIR-12 short text conversation tasks. In these models, however, there is no dependency on the history of conversation to determine the user's intent.

3. OUR APPROACH

We model the task of providing appropriate chat responses as an Information Retrieval problem, where for a given user message M and context C, the system retrieves and ranks the candidates by relevance and outputs one of the highest scoring responses. Offline, we create an index of paired tweets and their responses and index these using Lucene[1]. At runtime, the best response is chosen in a three step process. First, we use TF-IDF-based fetch to generate a candidate set appropriate to M and C. Then we extract features using a convolutional deep structured semantic network. Finally, a ranker is trained on 3-turn twitter conversations using these features to select response R from the candidate set (described in sections 3.4 below). Our setting is similar to any traditional IR system, and our main focus is on extracting relevant features and improving the candidate set selection. The runtime process is depicted in Figure 2.

[1]http://lucene.apache.org/

3.1 Data for Index and Ranker

3.1.1 *Data preparation for M-R pair Index*

We constructed a dataset of 17.62 million tweet conversational pairs (tweets and their responses), extracted from the Twitter Firehose, covering the four year period from 2012 through 2015. To favor responses reflecting a culturally local persona, we limited the geographical region to a specific time zone. This permitted us to expose more culturally-appropriate responses, for example, the query *what do you like for dinner* triggers the response *bhindi masala* (an Indian curry made with okra) for Indian users and *kaeng lueang ("yellow curry")* for users in Thailand.

In order to protect privacy and prevent personal information from surfacing in our chat agent's responses, we removed from this dataset, any conversational pairs where the response contained any individual's name, URL or hashtag. Further, we sought to minimize the risk of offending users by removing any pairs in which either M or R contained adult, politically sensitive, or ethnic-religious content, or other potentially offensive or contentious material, such as inappropriate references to violence, crime and illegal substances. We also filtered data to avoid scenarios where an adversarial query would result in retrieval of a candidate with high confidence from index like *Does <celebrity> hate <religion>? –> Yes, he does*, where <celebrity> and <religion> stand in for specific entities[2]. After filtering, our corpus comprises 9.56 million usable M-R pairs.

3.1.2 *Training Data for the Ranker*

To train the ranker, we obtained a 2.58 million sample of 3-turn tweet conversations from the Twitter Firehose, in the same manner as described in Section 3.1.1 above. Each conversation comprises an exchange between two Twitter users, alternating over 3 turns.

3.2 Retrieving Candidates

The goal of the retrieval step is to fetch, with high recall, relevant documents present in the index for the given M and C. As noted above, we have built an index of messages and responses. Given M, this index is used to retrieve best matching messages and their corresponding responses that constitute the candidate set. From a retrieval perspective, however, many messages in conversational scenarios are not self-sufficient as queries to return a good candidate set, but are dependent on expressions in the previous context e.g. *why?*, go ahead, and *please*. For shorter queries such as these, we fetch documents after appending context to query. This solution, though simple, works pragmatically and does not form the main focus of this paper.

3.3 Convolutional DSSM

To extract features for our ranker, we trained a Convolutional Deep Structured Semantic Model (cDSSM) [12]. cDSSM finds semantic relevance between text query and document, after being trained on click-through data to maximize conditional likelihood of the clicked documents for the given query. We adapted this formulation to our purpose by utilizing M-R pairs from Twitter (described in 3.1.1 above) as training data, where the user message is treated as query

[2]Queries involving such potentially offensive material must be detected and handled in a separate component that is beyond the scope of this paper.

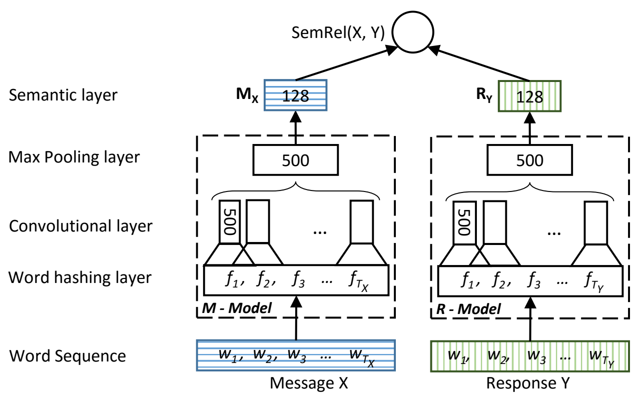

Figure 3: cDSSM uses a pair of models to vectorize message and response to their respective low-dimensional vectors.

and response as clicked document. The cDSSM model thus learns the semantic relevance between a message and its response. Note that semantic relevance indicates, for a given M-R pair, how relevant R is for M, and is distinct from semantic similarity. We chose cDSSM over DSSM[3] [3] in order to incorporate structural (word position) features in both message and response. To prepare the character trigram set [3], we took the most frequent 5k character trigrams found in the Twitter dataset.

Figure 3 shows two models, a Message model (M-Model) and a Response model (R-Model), that have been obtained after training on M-R pairs. When a text X is forward propagated through either of the models, it is vectorized in the space defined by that model. We use the notation $\mathbf{M_X}$ for the vector when X is vectorized using the M-model and $\mathbf{R_X}$ when it is vectorized using the R-Model. We define two scoring functions using the two models as follows:

Semantic Relevance Score(X,Y): We denote this score as *SemRel(X, Y)*. This is the confidence of semantic relevance of Y as a response to X, calculated as follows:

$$SemRel(X,Y) = cosine(\mathbf{M_X}, \mathbf{R_Y}) = \frac{\mathbf{M_X}^{\mathbf{T}}\mathbf{R_Y}}{\|\mathbf{M_X}\|\|\mathbf{R_Y}\|}$$

Semantic Similarity Score(X,Y): This score is denoted as *SemSim(X, Y)*. It indicates semantic similarity of two texts X and Y in the space defined by the model used. To compute semantic similarity of two responses X and Y, both X and Y are vectorized using the R-Model, using the following formula:

$$SemSim(X,Y) = cosine(\mathbf{R_X}, \mathbf{R_Y}) = \frac{\mathbf{R_X}^{T}\mathbf{R_Y}}{\|\mathbf{R_X}\|\|\mathbf{R_Y}\|}$$

The semantic similarity of two messages could also be computed using the M-model in an analogous manner, though we did not use this approach here. We use the above two scoring functions to generate features as described in Section 3.4 below.

[3]Initial experimentation with DSSM failed to yield useful results.

3.4 Learning to Rank in Multi-Turn Chat

Since we have modeled the problem as ranking task, we trained an implementation of the MART gradient boosting algorithm [1] to order the responses in candidate set. Note that as cDSSM is trained on only M-R pairs and thus learns to rank candidates only for a given message, we use it to derive features for the ranker to pick the best contextual response in the ongoing conversation.

Data format for ranker training: We used 3-turn Twitter conversations as described in Section 3.1.2. From each conversation, we prepared three training samples (1 positive and 2 negative), where Turn 1 and 2 were taken as C and M respectively. For positive samples, the original actual response in Turn 3 was used. For the two negative samples, random responses were selected (without replacement) from entire set of Tweets.

Features for ranker training: We used the following features to train the ranker, which incorporated context while ranking the candidates:
i) SemRel(M,R) – this represents the relevance of candidate response R for a given M without considering context. This is calculated from the cDSSM model as described in Section 3.3.
ii) Context Message Match (CMM) – These are the exact matches between C, M and R (borrowed from [13]). These features incorporate lexical similarity of R with M and C. As described in [13], we calculate the number of [1-4] n-gram matches between C and R, and between M and R. These matches are helpful for examples where for message *i love you*, *i love you too!* is a relevant response.
iii) SemSim(C,R) – This feature captures semantic similarity between C and R, calculated using cDSSM. It helps pick a response that is semantically closer to the context and hence capture the mood or intent of conversation. We discuss it further in Section 4.3 with an example.
iv) SemRel(C,M) – This feature captures change of context in the ongoing discussion. The score shows the relevance of M being the response for the C, where C was the previous message. The lower the value of this feature, the more likely it is that the user has changed topic and contextual features need not to be assigned higher importance.

We refer to the last two features *(iii) and (iv)* jointly as **Context Capturing Features (CCF)**. At training time, our best system with all defined features scored NDCG@1 = 84.83, NDCG@2 = 92.56 and NDCG@3 = 94.02 on our development evaluation set.

4. EVALUATION

4.1 Setup

Since we are primarily interested in the contextual appropriateness of interactions from a human viewpoint, we evaluated two versions of our system, SemRel(M,R) +CMM and SemRel(M,R)+CMM+CCF, using human relevance judgments. To this end, we held out 1000 3-turn conversations from our Twitter dataset. We took first 2 turns of these 1000 conversations and retrieved the third turn using our system. The retrieved response was shown to human judges and labelled for relevance, using a crowdsourcing platform. In each case, we employed 5 human judges, who were shown

Result	Context (C) and Message (M)	Response (R)	Comment
Relevant	**C:** *i want to go swimming!* **M:** *i was just thinking the same thing!*	*great minds think alike*	Model successfully retrieved relevant response on the basis of *SemRel(M,R)* for a non-context dependent M.
	C: *no! he is awful.* **M:** *he is? how so?*	*he's in critical condition*	Model was successful in retrieving relevant response by incorporating context. *SemSim(C,R)* captured semantic closeness between C and R due to words *critical condition* and *awful*.
Irrelevant	**C:** *may i know where do you from?* **M:** *i'm from indonesia. you?*	*i mean indonesia?*	Model was not able to separate *you?* as question in focus, and rather used a CMM feature (M-R match).

Table 1: Relevant and irrelevant outputs from the SemRel(M,R)+CMM+CCF system, with annotations concerning analysis

Technique	Score	95% CI
IR$_{status}$	3.73	3.67 3.79
IR$_{status}$+CMM	3.78	3.73 3.85
DCGM-II+CMM	**3.87**	3.80 3.95
SemRel(M,R)+CMM	4.15	4.10 4.21
SemRel(M,R)+CMM+CCF	**4.23**	4.17 4.29

Table 2: Mean scores assigned by judges, together with 95% confidence intervals. The best-performing baseline and experimental systems are shown in bold.

Technique	P@1	95% CI
IR$_{status}$	62.4	59.4 65.4
IR$_{status}$+CMM	63.8	60.8 67.8
DCGM-II+CMM	**68.4**	65.5 71.3
SemRel(M,R)+CMM	79.3	76.8 81.8
SemRel(M,R)+CMM+CCF	**82.0**	79.6 84.4

Table 3: Precision@1 together with 95% confidence intervals. The best-performing baseline and experimental systems are shown in bold.

the previous 2 turns by way of context along with the extracted response, and then asked to make a binary decision [0,1] whether or not the response was relevant to the conversational context. Other characteristics that might be appropriate in a human-agent interaction, such as interestingness and friendliness were not evaluated at this time. Also, since this is a retrieval-based system, we did not evaluate for grammatical correctness.

We compared two models against 3 baselines: IR$_{status}$(as in [8]), IR$_{status}$+CMM$_{10feat.}$, and DCGM-II+CMM$_{10feat.}$ (as proposed in [13]). For the purpose of our task, we believe the last is a strong baseline for a retrieval based system, since [13] also incorporated context when ranking candidates. When presenting the outputs to the judges, we randomly interpolated the system outputs to prevent the introduction of bias into the judgments.

4.2 Results

We report the results of human judgments in two ways. First, we compute scores for each of the 1000 responses on a 5 point scale: we convert the binary decisions into a score between 0 and 5 by summing the judgments, then average over those scores. The results are shown in Table 2. Two-tailed t-tests indicate that the difference in means between the top-scoring baseline (DCGM-II+CMM) and the two SemRel(M,R) systems is statistically significant ($p < 0.0001$). The SemRel(M,R)+CMM+CCF appears to be slightly better overall; CCF provides a small but useful boost.

User:	*hi, good evening*
Bot:	*hey..good evening..*
User:	*how are you*
Bot:	*i'm fine and you?*
User:	*i am also good, just returned from office*
Bot:	*where do you work?*
User:	*all that later, i have something to tell you*
Bot:	*what?*
User:	*i like a girl a lot, and want to ask her out*
Bot:	*ask her out anyway*

Figure 4: A sample user interaction with our system illustrating the human-like nature of exchanges.

To assess the precision of retrieval, we also used a more conservative measure. Responses that achieved a supermajority of votes (≥ 4) were assigned a score of 1, otherwise they were assigned 0. This we will term *Precision@1*, in other words, precision at rank 1. The *Precision@1* results are presented in Table 3. It is evident from the confidence intervals that using features from deep learning models (i.e., cDSSM) to capture deep structured similarity yields significant improvement over other methods.

The gain in precision between SemRel(M,R)+CMM and the DCGM-II+CMM baseline is particularly significant ($p < 0.0001$) as measured by the Wilcoxon signed-rank test and McNemar's test for dichotomous variables. The addition of CCF features (SemRel(M,R)+CMM+CCF) appears to play a useful role in capturing context and in fine tuning ranking according to context.

4.3 Qualitative Analysis

Table 1 shows examples of relevant and irrelevant results returned by the SemRel(M,R)+CMM+CCF model. The examples are taken from our 1000 conversation evaluation set as described in Section 4. The top example in the relevant results section is an instance where we observe same response with or without CCF features. We also note that there are no n-gram matches in C and R. This implies that *SemRel(M,R)* was alone sufficient to retrieve a relevant response, because M itself is descriptive and not heavily dependent on C. The example also suggests that our model does not overweight contextual features.

The second example in the relevant results, shows that the *SemSim(C,R)* feature helps retrieve a relevant response that is consistent with C. Candidate responses for M (*he is? how so?*) included *who he is ?* and *aw tell me about it!! he is so hot!*, which are relevant to M but irrelevant when C is taken into account.

The irrelevant result shown in Table 1 illustrates a known limitation of our model. Here, an irrelevant response was retrieved because *you?* was the focus of the message: candidates should have been ranked taking into account *you?* as M and *i am from indonesia.* as C. It would appear that none of the candidates received a high score for SemRel(M,R), while the CMM feature for M-R lexical match was assigned high importance.

Figure 4 provides a short sample of chat by a human using our system, illustrating the fluency and human-like nature of exchanges.

5. CONCLUSIONS

We have presented initial results of experiments in the use of convolutional Deep Structured Semantic neural networks (cDSSM) to emulate human conversations using IR techniques. Our approach is not only computationally performant but produces results that are significantly more relevant than baseline IR techniques. In these experiments, we employed a simple feature set and focused primarily on learning from cDSSM to output the best responses in context. In the future, we hope to expand the candidate set by introducing better context understanding and by taking into account user sentiment in order to further improve the quality of the human-agent interaction. Our results suggest that cDSSM is a viable approach to emulate interesting conversational exchanges, especially in cases where conversational data may be relatively limited, for example, regional markets and "smaller" or minority languages.

6. ACKNOWLEDGMENTS

We thank Bill Dolan, Michel Galley, Jianfeng Gao, Manoj K Chinnkotla, Chakrapani Ravi Kiran, Puneet Garg and Radhakrishnan Srikanth for helpful discussions and comments. We would also like to thank the two anonymous reviewers for their comments.

7. REFERENCES

[1] C. J. Burges. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11:23–581, 2010.

[2] H. Denawa, T. Sano, Y. Kadotami, S. Kato, and T. Sakai. SLSTC at the NTCIR-12 STC task. In *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies*, 2016.

[3] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of CIKM*, pages 2333–2338. ACM, 2013.

[4] S. Jafarpour, C. J. Burges, and A. Ritter. Filter, rank, and transfer the knowledge: Learning to chat. *Advances in Ranking*, 10, 2010.

[5] J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan. A diversity-promoting objective function for neural conversation models. In *Proceedings of HLT-NAACL*, forthcoming 2016.

[6] Y. Liu, C. Sun, L. Lin, and X. Wang. ITNLP: Pattern-based short text conversation system at NTCIR-12. In *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies*, 2016.

[7] X. Qiu and X. Huang. Convolutional neural tensor network architecture for community-based question answering. In *Proceedings of IJCAI*, pages 1305–1311, 2015.

[8] A. Ritter, C. Cherry, and W. B. Dolan. Data-driven response generation in social media. In *Proceedings of EMNLP*, pages 583–593. ACL, 2011.

[9] A. Severyn and A. Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of SIGIR*, pages 373–382. ACM, 2015.

[10] L. Shang, Z. Lu, and H. Li. Neural responding machine for short-text conversation. In *Proceedings of ACL*, pages 1577–1586, 2015.

[11] L. Shang, T. Sakai, Z. Liu, H. Li, R. Higashinaka, and Y. Miyao. Overview of the NTCIR-12 short text conversation task. In *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies*, 2016.

[12] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of WWW*, pages 373–374, 2014.

[13] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J.-Y. Nie, J. Gao, and B. Dolan. A neural network approach to context-sensitive generation of conversational responses. In *NAACL-HLT*, pages 196–205, 2015.

[14] H. Sugiyama. Utterance selection based on sentence similarities and dialogue breakdown detection on NTCIR-12 STCV task. In *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies*, 2016.

[15] R. Swanson and A. S. Gordon. Say anything: A massively collaborative open domain story writing companion. In *Interactive Storytelling*, pages 32–40. Springer, 2008.

[16] O. Vinyals and Q. Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.

[17] W.-t. Yih, X. He, and C. Meek. Semantic parsing for single-relation question answering. In *ACL*, pages 643–648, 2014.

[18] P. Yin, Z. Lu, H. Li, and B. Kao. Neural enquirer: Learning to query tables with natural language, 2015.

[19] L. Yu, K. M. Hermann, P. Blunsom, and S. Pulman. Deep learning for answer sentence selection. *NIPS deep learning workshop*, 2014.