A Neural Network Approach to Context-Sensitive Generation of Conversational Responses

Alessandro Sordoni[*1] **Michel Galley**[2] **Michael Auli**[*3] **Chris Brockett**[2]
Yangfeng Ji[*4] **Meg Mitchell**[2] **Jian-Yun Nie**[1] **Jianfeng Gao**[2] **Bill Dolan**[2]
[1]DIRO, Université de Montréal, Montréal, QC, Canada
{sordonia, nie}@iro.umontreal.ca
[2]Microsoft Research, Redmond, WA, USA
{mgalley,chrisbkt,memitc,jfgao,billdol}@microsoft.com
[3]Facebook AI Research, Menlo Park, CA, USA
michaelauli@fb.com
[4]Georgia Institute of Technology, Atlanta, GA, USA
jiyfeng@gatech.edu

Abstract

We present a novel response generation system that can be trained from end to end on large quantities of unstructured Twitter conversations. A neural network architecture is used to address sparsity issues that arise when integrating context information into classical statistical models, allowing the system to take into account previous dialog utterances. Our dynamic-context generative models show consistent gains over both context-sensitive and non-context-sensitive Machine Translation and Information Retrieval baselines.

1 Introduction

Until recently, the goal of training open-domain conversational systems that naturally mimic human conversation has seemed elusive. However, the vast quantities of conversational exchanges now available on social media websites such as Twitter and Reddit raise the prospect of building data-driven models that might eventually be integrated into larger, task-oriented systems. The work of (Ritter et al., 2011), for example, demonstrates that a response generation system can be constructed from Twitter conversations using statistical machine translation techniques, where a status post by a Twitter user is "translated" into a plausible looking response.

The statistical machine translation model presented in (Ritter et al., 2011) cannot easily address the challenge of generating responses in context.

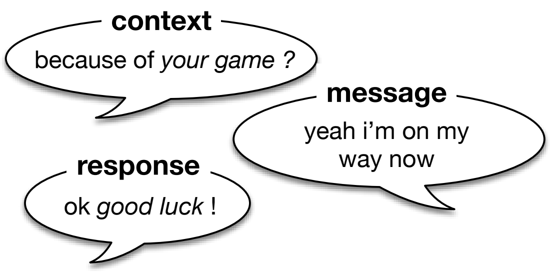

Figure 1: Example of three consecutive utterances occurring between two Twitter users A and B.

Broadly speaking, context may be linguistic or involve grounding in the physical or virtual world, but we here focus exclusively on linguistic context. The ability to take into account previous utterances is key to building dialog systems that can keep conversations active and engaging. Figure 1 illustrates a typical Twitter dialog where the contextual information is crucial: the phrase "good luck" is plainly motivated by the reference to "your game" in the first utterance. In the MT model, such contextual sensitivity is difficult to capture; moreover, naive injection of context information would entail unmanageable growth of the phrase table at the cost of increased sparsity, and skew towards rarely-seen context pairs. In most statistical approaches to machine translation, phrase pairs do not share statistical weights regardless of their intrinsic semantic commonality (Kalchbrenner and Blunsom, 2013; Gao et al., 2014).

We propose to address the challenge of context-sensitive response generation by using continuous representations or *embeddings* of words and phrases to compactly encode semantic and syntactic similar-

*The entirety of this work was conducted while at Microsoft Research.

ity. We argue that embedding-based models afford flexibility to model the transitions between consecutive utterances and to capture long-span dependencies in a domain where traditional word and phrase alignment is difficult (Ritter et al., 2011) and to this end we present two simple, context-sensitive response-generation models utilizing the Recurrent Neural Network Language Model (RLM) architecture (Mikolov et al., 2010). These models first encode past information in a hidden continuous representation, which is then decoded by a RLM to promote plausible responses that are simultaneously fluent and contextually relevant. Unlike typical complex task-oriented multi-modular dialog systems (Young, 2002; Stent and Bangalore, 2014), our architecture is *completely data-driven* and can easily be trained end-to-end using unstructured data without requiring human annotation, scripting, or automatic parsing.

This paper makes the following contributions. We present a neural network architecture for response generation that is both context-sensitive and data-driven. As such, it can be trained from end to end on massive amounts of social media data. To our knowledge, this is the first application of a neural-network model to open-domain response generation, and we believe that the present work will lay in vital groundwork for more complex models to come. Finally, we introduce a novel multi-reference extraction technique that shows promise for automated evaluation.

2 Related Work

Our work naturally lies in the path opened by (Ritter et al., 2011), but we generalize their approach by exploiting information from a larger context. Ritter et al. and our work represent a radical paradigm shift from other work in dialog. More traditional dialog systems typically tease apart dialog management (Young, 2002) from response generation (Stent and Bangalore, 2014), while our holistic approach can be considered as a first attempt to accomplish both tasks jointly. While there are previous uses of machine learning for response generation (Walker et al., 2003), dialog state tracking (Young et al., 2010), dialog and user modeling (Georgila et al., 2006), many components of typical dialog systems remain hand coded, in particular the labels and attributes defining dialog states. In contrast, the dialog state in our neural network model is completely latent and directly optimized towards end-to-end performance. In this sense, we believe the framework of this paper is a significant milestone towards more data-driven and less hand-coded dialog processing.

Continuous representations of words and phrases estimated by neural network models have been applied on a variety of tasks ranging from Information Retrieval (IR) (Huang et al., 2013), Machine Translation (MT) (Auli et al., 2013; Cho et al., 2014; Sutskever et al., 2014) and Language Modeling (LM) (Bengio et al., 2003; Collobert and Weston, 2008). Gao et al. (2014) successfully use an embedding model to refine the estimation of rare phrase-translation probabilities, which is classically affected by sparsity problems. Robustness to sparsity is a crucial property for our task as it allows to capture context information while avoiding unmanageable growth of model parameters. Our work extends the Recurrent Neural Network Language Model (RLM) (Mikolov et al., 2010) which uses continuous representations to estimate a probability function over natural language sentences. We propose a set of conditional RLMs where contextual information (i.e., past utterances) is encoded in a continuous context vector to help generate the response. Our models differ from most previous work in the way the context vector is constructed. For example, (Mikolov and Zweig, 2012; Auli et al., 2013) use a pre-trained topic model. In our models, the context vector is learnt along with the conditional RLM that generates the response. The learnt context encodings do not exclusively capture contentful words. Even "stop words" can carry discriminative power – consider the response to "how are you ?".

3 Recurrent Language Model

We give a brief overview of the Recurrent Language Model (RLM) (Mikolov et al., 2010) architecture which our models extend. A RLM is a generative model of sentences, i.e. given a sentence $s = s_1, \dots, s_T$, it estimates:

$$p(s) = \prod_{t=1}^{T} p(s_t | s_1, \dots, s_{t-1}). \qquad (1)$$

The model architecture is parameterized by three weight matrices, $\Theta_{\text{RNN}} = \langle W_{in}, W_{out}, W_{hh} \rangle$: an input matrix W_{in}, a recurrent matrix W_{hh} and an output matrix W_{out}, which are usually initialized randomly. The rows of the input matrix $W_{in} \in \mathbb{R}^{V \times K}$ contain the K-dimensional embeddings for each word in the language vocabulary of size V. Let us denote by s_t both the vocabulary token and its one-hot representation, i.e. a zero vector of dimensionality V with a 1 corresponding to the index of the s_t token. The embedding for s_t is then obtained by $s_t^\top W_{in}$. The recurrent matrix $W_{hh} \in \mathbb{R}^{K \times K}$ keeps a history of the subsequence that has already been processed. The output matrix $W_{out} \in \mathbb{R}^{K \times V}$ projects the hidden state h_t into the output layer o_t which has an entry for each word in the vocabulary V. This value is used to generate a probability distribution for the next word in the sequence. Specifically, the forward pass proceeds with the following recurrence, for $t = 1, \ldots, T$:

$$h_t = \sigma(s_t^\top W_{in} + h_{t-1}^\top W_{hh}), \quad o_t = h_t^\top W_{out} \tag{2}$$

where σ is a non-linear function applied element-wise, in our case the logistic sigmoid. The recurrence is seeded by setting $h_0 = 0$, the zero vector. The probability distribution over the next word given the previous history is obtained by applying the softmax activation function:

$$P(s_t = w | s_1, \ldots, s_{t-1}) = \frac{\exp(o_{tw})}{\sum_{v=1}^{V} \exp(o_{tv})}. \tag{3}$$

The RLM is trained to minimize the negative log-likelihood of the training sentence s:

$$L(s) = -\sum_{t=1}^{T} \log P(s_t | s_1, \ldots, s_{t-1}). \tag{4}$$

The recurrence is unrolled backwards in time using the back-propagation through time (BPTT) (Rumelhart et al., 1988) algorithm and gradients are accumulated over multiple time-steps.

4 Context-Sensitive Models

We distinguish three linguistic entities in a conversation between two users A and B: the context[1] c, the message m and response r. The context c represents a sequence of past dialog exchanges of any length; then B emits a message m to which A reacts by formulating its response r (see Figure 1).

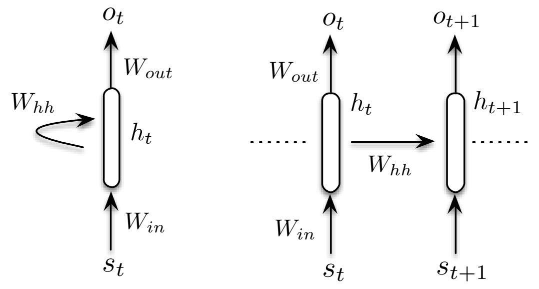

Figure 2: Compact representation of an RLM (left) and unrolled representation for two time steps (right).

We use three context-based generation model to estimate a generation model of the response r, $r = r_1, \ldots, r_T$, conditioned on past information c and m:

$$p(r|c, m) = \prod_{t=1}^{T} p(r_t | r_1, \ldots, r_{t-1}, c, m). \tag{5}$$

These three models differ in the manner in which they compose the context-message pair (c, m).

4.1 Tripled Language Model

In our first model, dubbed RLMT, we straightforwardly concatenate each utterance c, m, r into a single sentence s and train the RLM to minimize $L(s)$. Given c and m, we compute the probability of the response as follows: we perform the forward propagation over the known utterances c and m to obtain a hidden state encoding useful information about previous utterances. Subsequently, we compute the likelihood of the response from that hidden state.

An issue with this simple approach is that the concatenated sentence s will be very long on average, especially if the context comprises multiple utterances. Modelling such long-range dependencies with an RLM is difficult and is still considered an open problem (Pascanu et al., 2013). We will consider RLMT as an additional context-sensitive baseline for the models we present next.

4.2 Dynamic-Context Generative Model I

The above limitation of RLMT can be addressed by strengthening the context bias. In our second model (DCGM-I), the context and the message are encoded

[1]In this work, the context is purely linguistic but interesting future work might integrate other types of context information, e.g. geographical location or time information, or other forms of grounding.

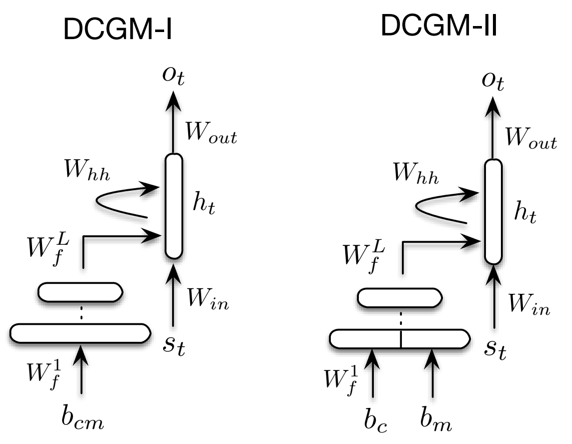

Figure 3: Compact representations of DCGM-I (left) and DCGM-II (right). The hidden state of the decoder RLM receives a bias from the context encoder. In DCGM-I, we encode the bag-of-words representation of both c and m in a single vector b_{cm}. In DCGM-II, we concatenated the representations b_c and b_m on the first layer to preserve order information.

into a fixed-length vector representation which is used by the RLM to decode the response. This is illustrated in Figure 3 (left). First, we consider c and m as a single sentence and compute a single bag-of-words representation $b_{cm} \in \mathbb{R}^V$. Then, b_{cm} is provided as input to a multilayered non-linear forward architecture that produces a fixed-length representation that is used to bias the recurrent state of the decoder RLM. At training time, both the context encoder and the RLM decoder are learned so as to minimize the negative log-probability of the generated response.

Formally, the parameters of the model are $\Theta_{\text{DCGMI}} = \langle W_{in}, W_{hh}, W_{out}, \{W_f^\ell\}_{\ell=1}^L \rangle$, where $\{W_f^\ell\}_{\ell=1}^L$ are the weights for the L layers of the feed-forward context networks. The fixed-length context vector k_L is obtained by a forward propagation of the network:

$$\begin{aligned} k_1 &= b_{cm}^\top W_f^1 \\ k_\ell &= \sigma(k_{\ell-1}^\top W_f^\ell) \quad \text{for } \ell = 2, \cdots, L \end{aligned} \tag{6}$$

The rows of W_f^1 contain the embeddings of the vocabulary.[2] These are different from those used in the RLM and play a crucial role in promoting the specialization of the context encoder to a distinct task. The hidden layer of the decoder RLM takes the following form:

$$h_t = \sigma(h_{t-1}^\top W_{hh} + k_L + s_t^\top W_{in}) \tag{7a}$$

$$o_t = h_t^\top W_{out} \tag{7b}$$

$$p(s_{t+1}|s_1, \ldots, s_{t-1}, c, m) = \text{softmax}(o_t) \tag{7c}$$

This model conditions on the previous utterances via biasing the hidden layer state on the context representation k_L. Note that the context representation does not change through time. This is useful because: (a) it forces the context encoder to produce a representation general enough to be useful for generating all words in the response and (b) it helps the RLM decoder to remember context information when generating long responses.

4.3 Dynamic-Context Generative Model II

Because DCGM-I does not distinguish between c and m, that model has the propensity to underestimate the strong dependency that holds between m and r. Our third model (DCGM-II) addresses this issue by concatenating the two linear mappings of the bag-of-words representations b_c and b_m in the input layer of the feed-forward network representing c and m (see Figure 3 right). Concatenating continuous representations prior to deep architectures is a common strategy to obtain order-sensitive representations (Bengio et al., 2003; Devlin et al., 2014).

Formally, the forward equations for the context-encoder write as:

$$\begin{aligned} k_1 &= [b_c^\top W_f^1, b_m^\top W_f^1], \\ k_\ell &= \sigma(k_{\ell-1}^\top W_f^\ell) \quad \text{for } \ell = 2, \cdots, L \end{aligned} \tag{8}$$

where $[x, y]$ denotes the concatenation of x and y vectors. Also in DCGM-II, the bias on the recurrent hidden state and the probability distribution over the next token are computed as described in Eq. 7.[3]

5 Experimental Setting

5.1 Dataset Construction

For computational efficiency and to alleviate the burden of human evaluators, we restrict the context se-

[2]Notice that the first layer of the encoder network is linear. We found that this helps learning the embedding matrix as it reduces the vanishing gradient effect partially due to stacking of squashing non-linearities (Pascanu et al., 2013).

[3]An implementation of these models will be made available.

Corpus	# Triples	Avg # Ref	[Min,Max] # Ref
Tuning	2118	3.22	[1, 10]
Test	2114	3.58	[1, 10]

Table 1: Number of triples, average, minimum and maximum number of references for tuning and test corpora.

quence c to a single sentence. Hence, our dataset is composed of "triples" $\tau \equiv (c_\tau, m_\tau, r_\tau)$ consisting of three sentences. We mined 127M context-message-response triples from the Twitter FireHose, covering the 3-month period June 2012 through August 2012. Only those triples where context and response were generated by the same user were extracted. To minimize noise, we selected triples that contained at least one frequent bigram that appeared more than 3 times in the corpus. This produced a corpus of 29M Twitter triples. In addition, 2118 tuning set triples and 2114 test set triples[4] were sub-selected from a candidate set of approximately 33K candidates after quality vetting by crowd-sourced raters. Judgments on a 5 point scale were received from 3 raters apiece, and only triples with an average 4 or better were selected for inclusion.

5.2 Automatic Evaluation

We evaluate all systems using BLEU (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005), and supplement these results with more targeted human pairwise comparisons in Section 6.3. A major challenge in using these automated metrics for response generation is that the set of reasonable responses in our task is potentially vast and extremely diverse. The dataset construction method just described yields only a single reference for each status. Accordingly, we extend the set of references using an IR approach to mine potential responses, after which we have human judges rate their appropriateness. We found that optimizing systems towards BLEU using the mined multi-references aligned BLEU rankings surprisingly well with human judgments, as confirmed in Section 6.3. This lays ground work for interesting future correlation studies.

Multi-reference extraction We use the following algorithm to better cover the space of reasonable responses. Given a test triple $\tau \equiv (c_\tau, m_\tau, r_\tau)$, our goal is to mine other responses $\{r_{\tilde{\tau}}\}$ that fit the context and message pair (c_τ, m_τ). To this end, we first select a set of 15 candidate triples $\{\tilde{\tau}\}$ using an IR system. The IR system is calibrated in order to select candidate triples $\tilde{\tau}$ for which both the message $m_{\tilde{\tau}}$ and the response $r_{\tilde{\tau}}$ are similar to the original message m_τ and response r_τ. Formally, the score of a candidate triple is:

$$s(\tilde{\tau}, \tau) = d(m_{\tilde{\tau}}, m_\tau)\,(\alpha\, d(r_{\tilde{\tau}}, r_\tau) + (1-\alpha)\epsilon), \quad (9)$$

where d is the bag-of-words BM25 similarity function (Robertson et al., 1995), α controls the impact of the similarity between the responses and ϵ is a smoothing factor that avoids zero scores for candidate responses that do not share any words with the reference response. We found that this simple formula provided references that were both diverse and plausible. Given a set of candidate triples $\{\tilde{\tau}\}$, human evaluators are asked to rate the quality of the response within the new triples $\{(c_\tau, m_\tau, r_{\tilde{\tau}})\}$. After human evaluation, we retain the references for which the score is better or equal than 4 on a 5 point scale, resulting in 3.58 references per example on average (Table 1).

5.3 Feature Sets

The response generation systems evaluated in this paper are parameterized as log-linear models in a framework typical of statistical machine translation (Och and Ney, 2004). These log-linear models comprise the following feature sets:

MT MT features are derived from a large response generation system built along the lines of (Ritter et al., 2011), which is based on a phrase-based MT decoder similar to Moses (Koehn et al., 2007). Our MT feature set includes the following features that are common in Moses: forward and backward maximum likelihood "translation" probabilities, word and phrase penalties, linear distortion, and a modified Kneser-Ney language model (Kneser and Ney, 1995) trained on Twitter responses. For the translation probabilities, we built a very large phrase table of 160.7 million entries by first filtering out Twitterisms (e.g., long sequences of vowels, hashtags), and then selecting candidate phrase pairs using Fisher's exact test (Ritter et al., 2011). We also included MT de-

[4]We are working to be able to release the ids of these triples.

System	BLEU
RANDOM	0.33
MT	3.21
HUMAN	6.08

Table 2: Multi-reference corpus-level BLEU obtained by leaving one reference out at random.

coder features specifically motivated by the response generation task: Jaccard distance between source and target phrase, Fisher's exact probability, and a score relating the lengths of source and target phrases.

IR We also use an IR feature built from an index of triples, whose implementation roughly matches the $\text{IR}_{\text{status}}$ approach described in (Ritter et al., 2011): for a test triple τ we chose $r_{\tilde{\tau}}$ as the candidate response iff $\tilde{\tau} = \arg\max_{\tilde{\tau}} d(m_\tau, m_{\tilde{\tau}})$.

CMM Neither MT nor IR take into account context information. Therefore, we take into consideration context and message matches (CMM), i.e. exact matches between c, m and r. We define 8 features as the [1-4]-gram matches between c and the candidate reply r and the [1-4]-gram matches between m and the candidate reply r. These exact matches help capture and promote context information in the replies.

RLMT, DCGM-I, DCGM-II We consider the RLM trained on the concatenated triples, denoted as RLMT (Section 4.1), to be a context-sensitive RLM baseline. Each neural network model contributes an additional feature corresponding to the likelihood of the candidate response given the context and the message.

5.4 Model Training

The proposed models are trained on a 4M subset of the triple data. The vocabulary consists of the most frequent $V = 50K$ words. In order to speed up training, we use the Noise-Contrastive Estimation (NCE) loss which avoids repeated summations over V by approximating the probability of the target word (Gutmann and Hyvärinen, 2010). Parameter optimization is done using Adagrad (Duchi et al., 2011) with a mini-batch size of 100 and a learning rate $\alpha = 0.1$, which we found to work well on held-out data. In order to stabilize learning, we clip the gradients to a fixed range $[-10, 10]$, as suggested in (Mikolov et al., 2010). All the parameters of the neural models are sampled from a normal distribution $\mathcal{N}(0, 0.01)$ while the recurrent weight W_{hh} is initialized as a random orthogonal matrix. To prevent over-fitting, we evaluate performance on a held-out set during training and stop when the objective increases. The size of the RLM hidden layer is set to $K = 512$, where the context encoder is a 512, 256, 512 multilayer network. The bottleneck in the middle compresses context information that leads to similar responses and thus achieves better generalization. The last layer embeds the context vector into the hidden space of the decoder RLM.

5.5 Rescoring Setup

We evaluate the proposed models by rescoring the n-best candidate responses obtained using the MT phrase-based decoder and the IR system. In contrast to MT, the candidate responses provided by IR have been created by humans and are less affected by fluency issues. The different n-best lists will provide a comprehensive testbed for our experiments. First, we augment the n-best list of the tuning set with the scores of the model of interest. Then, we run an iteration of MERT (Och, 2003) to estimate the log-linear weights of the new features. At test time, we rescore the test n-best list with the new weights.

6 Results

6.1 Lower and Upper Bounds

Table 2 shows the expected upper and lower bounds for this task as suggested by BLEU scores for human and a random baseline. The RANDOM system comprises responses randomly extracted from the triples corpus. HUMAN is computed by choosing one reference at random amongst the multi-reference set for each context-status pair.[5] Although the scores are lower than those usually reported in SMT tasks, the ranking of the three systems is unambiguous.

[5]For the human score, we compute corpus-level BLEU with a sampling scheme that randomly leaves out one reference - the human sentence to score - for each reference set. This sampling scheme (repeated with 100 trials) is also applied for the MT and RANDOM system so as to make BLEU scores comparable.

MT n-best	BLEU (%)	METEOR (%)	IR n-best	BLEU (%)	METEOR (%)
MT $_{9\text{ feat.}}$	3.60 *(-9.5%)*	9.19 *(-0.9%)*	IR $_{2\text{ feat.}}$	1.51 *(-55%)*	6.25 *(-22%)*
CMM $_{9\text{ feat.}}$	3.33 *(-16%)*	9.34 *(+0.7%)*	CMM $_{9\text{ feat.}}$	3.39 *(-0.6%)*	8.20 *(+0.6%)*
$\triangleright$ MT + CMM $_{17\text{ feat.}}$	3.98 (-)	9.28 (-)	$\triangleright$ IR + CMM $_{10\text{ feat.}}$	3.41 (-)	8.04 (-)
RLMT $_{2\text{ feat.}}$	4.13 *(+3.7%)*	9.54 *(+2.7%)*	RLMT $_{2\text{ feat.}}$	2.85 *(-16%)*	7.38 *(-8.2%)*
DCGM-I $_{2\text{ feat.}}$	4.26 *(+7.0%)*	9.55 *(+2.9%)*	DCGM-I $_{2\text{ feat.}}$	3.36 *(-1.5%)*	7.84 *(-2.5%)*
DCGM-II $_{2\text{ feat.}}$	4.11 *(+3.3%)*	9.45 *(+1.8%)*	DCGM-II $_{2\text{ feat.}}$	3.37 *(-1.1%)*	8.22 *(+2.3%)*
DCGM-I + CMM $_{10\text{ feat.}}$	4.44 *(+11%)*	9.60 *(+3.5%)*	DCGM-I + CMM $_{10\text{ feat.}}$	4.07 *(+19%)*	8.67 *(+7.8%)*
DCGM-II + CMM $_{10\text{ feat.}}$	4.38 *(+10%)*	9.62 *(+3.5%)*	DCGM-II + CMM $_{10\text{ feat.}}$	4.24 *(+24%)*	8.61 *(+7.1%)*

Table 3: Context-sensitive ranking results on both MT (left) and IR (right) n-best lists, $n = 1000$. The subscript $_{\text{feat.}}$ indicates the number of features of the models. The log-linear weights are estimated by running one iteration of MERT. We mark by ($\pm$%) the relative improvements with respect to the reference system ($\triangleright$).

6.2 BLEU and METEOR

The results of automatic evaluation using BLEU and METEOR are presented in Table 3, in which some broad patterns can be discerned. First, both metrics indicate that a phrase-based MT decoder outperforms a purely IR approach. Second, adding CMM features to the baseline systems helps. Third, the neural network models contribute measurably to improvement: RLMT and DCGM models outperform baselines, and DCGM models provide more consistent gains than RLMT.

MT vs IR BLEU and METEOR scores indicate that the phrase-based MT decoder outperforms a purely IR approach, despite the fact that IR proposes fluent human generated responses. This may be because the IR model only loosely captures important patterns between message and response: it ranks candidate responses solely by the similarity of their message with the message of the test triple (§5.3). As a result, the top ranked response is likely to drift from the purpose of the original conversation. The MT approach, by contrast, more directly models statistical patterns between message and response.

CMM MT+CMM, totaling 17 features (9 from MT + 8 CMM) improves 0.38 BLEU points, a 9.5% relative improvement, over the baseline MT model. IR+CMM, 10 features (IR + word penalty + 8 CMM) benefits even more, attaining 1.8 BLEU points and 1.5 METEOR points over the IR baseline. Figure 4 (a) and (b) plots the magnitude of learned CMM feature weights for MT+CMM and IR+CMM. CMM features help in both these hypothesis spaces and especially on the IR n-best list. Figure 4 (b) supports the hypothesis formulated in the previous paragraph: as IR captures only inter-message similarities, the matches between message and response are particularly important and context matches still help in providing additional gains. On the contrary, the phrase-based statistical patterns captured by the MT system do a good job in explaining away 1-gram and 2-gram message matches (Figure 4 (a)) and the performance gain mainly comes from context matches.

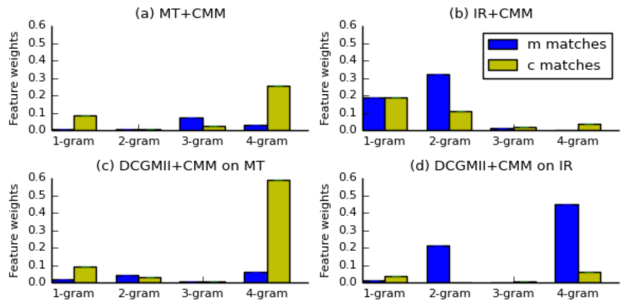

Figure 4: Comparison of the weights of learned CMM features for MT+CMM and IR+CMM systems (a) et (b) and DCGMII+CMM on MT and IR (c) and (d).

RLMT and DCGM Both RLMT and DCGM models outperform their respective MT and IR baselines. Both models also exhibit similar performance and show improvements over the MT+CMM models, albeit using a lower dimensional feature space. We believe that their similar performance is due to the limited diversity of MT n-best list together with gains in fluency stemming from the strong language model provided by the RLM. In the case of IR models, on the other hand, there is more headroom for

System A	System B	Gain (%)	CI
HUMAN	MT+CMM	13.6*	[12.4,14.8]
DCGM-II	MT	1.9*	[0.8, 2.9]
DCGM-II+CMM	MT	3.1*	[2.0, 4.3]
DCGM-II+CMM	MT+CMM	1.5*	[0.5, 2.5]
DCGM-II	IR	5.2*	[4.0, 6.4]
DCGM-II+CMM	IR	5.3*	[4.1, 6.6]
DCGM-II+CMM	IR+CMM	2.3*	[1.2, 3.4]

Table 4: Pairwise human evaluation scores between System A and B. The first (second) set of results refer to the MT (IR) hypothesis list. The asterisk means agreement between human preference and BLEU rankings.

improvement and fluency is already guaranteed. Any gains must come from context and message matches. Hence, RLMT underperforms with respect to both DCGM and IR+CMM. The DCGM models appear to have better capacity to retain contextual information and thus achieve similar performance to IR+CMM despite their lack of exact n-gram match information.

In the present experimental setting, no striking performance difference can be observed between the two versions of the DCGM architecture. If multiple sequences were used as context, we expect that the DCGM-II model would likely benefit more owing to the separate encoding of message and context.

DCGM+CMM We also investigated whether mixing exact CMM n-gram overlap with semantic information encoded by the DCGM models can bring additional gains. DCGM{I-II}+CMM systems each totaling 10 features show increases of up to 0.48 BLEU points over MT+CMM and up to 0.88 BLEU over the model based on (Ritter et al., 2011). METEOR improvements similarly align with BLEU improvements both for MT and IR lists. We take this as evidence that CMM exact matches and DCGM semantic matches interact positively, a finding that comports with (Gao et al., 2014), who show that semantic relationships mined through phrase embeddings coupled positively with classical co-occurrence-based estimations. Analysis of CMM feature weights in Figure 4, (c) and (d) suggests that 1-gram matches are explained away by the DCGM model, but that higher order matches are important. It appears that DCGM models might be improved by preserving word-order information in context and message encodings–a possible area of interesting future investigation.

6.3 Human Evaluation

Human evaluation was conducted using crowd-sourced annotators. Annotators were asked to compare the quality of system output responses pairwise (”which is better?”) in relation to the context and message strings in the 2114 item test set. Identical strings were held out, so that the annotators only saw those outputs that differed. Paired responses were presented in random order to the annotators and each pair of responses was judged by 5 annotators.

Table 4 summarizes the results of human evaluation, giving the difference in mean scores (pairwise preference margin) between systems and 95% confidence intervals generated using the Welch t-test. Identical strings not shown to raters are incorporated with an automatically assigned score of 0.5, indicating equivalence. The pattern in these results is clear and consistent: context-sensitive systems (+CMM) outperform non-context-sensitive systems, with preference gains as high as approximately 5.3% in the case of DCGM-II+CMM versus IR, and about 3.1% in the case of DCGM-II+CMM versus MT. Similarly, context-sensitive DCGM systems outperform non-DCGM context-sensitive systems by 1.5% (MT) and 2.3% (IR). These results are consistent with the automated BLEU rankings and confirm that our best performing DCGM models outperform both raw baseline and the context-sensitive baseline using CMM features.

7 Conclusion

We have formulated a neural network architecture for data-driven response generation trained from social media conversations. In distinction to previous work, the generation of the response is conditioned on past dialog utterances, considered as context information. We have proposed a novel multi-reference extraction technique allowing for robust automated evaluation using standard SMT metrics such as BLEU and METEOR. Our context-sensitive models consistently outperform both context-independent and context-sensitive baselines by up to 11% relative improvement in BLEU in the MT setting and 24% in the IR setting, albeit using a minimal amount of features. As our models are completely data-driven and self-contained, they could improve fluency and contextual relevance in other types of dialog systems.

Our work opens several directions for future research. We believe that there is much room for improvement by envisioning more complex neural network models that take into account word order within the message and context utterances. In addition, future progress in this area will be greatly enhanced by thorough study of automated evaluation measures.

References

Michael Auli, Michel Galley, Chris Quirk, and Geoffrey Zweig. 2013. Joint language and translation modeling with recurrent neural networks. In *Proc. of EMNLP*, pages 1044–1054.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proc. of ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Jun.

Yoshua Bengio, Rejean Ducharme, and Pascal Vincent. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *In Proc. of EMNLP*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proc. of ICML*, pages 160–167. ACM.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *Proc. of ACL*.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159.

Jianfeng Gao, Xiaodong He, Wen tau Yih, and Li Deng. 2014. Learning continuous phrase representations for translation modeling. In *Proc. of ACL*. Association for Computational Linguistics, June.

Kallirroi Georgila, James Henderson, and Oliver Lemon. 2006. User simulation for spoken dialogue systems: Learning and evaluation. In *Proc. of Interspeech/ICSLP*.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *International Conference on Artificial Intelligence and Statistics*, pages 297–304.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proc. of CIKM*, pages 2333–2338.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. *In Proc. of EMNLP*.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for M-gram language modeling. In *Proc. of the International Conference on Acoustics, Speech, and Signal Processing*, pages 181–184, May.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proc. of ACL Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, Jun.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Annual Conference of the International Speech Communication Association*, pages 1045–1048.

Franz Josef Och and Hermann Ney. 2004. The alignment template approach to machine translation. 30(4):417–449.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. pages 160–167, Sapporo, Japan, July.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. pages 311–318, Philadelphia, PA, USA, Jul.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. *Proc. of ICML*.

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*, EMNLP '11, pages 583–593, Stroudsburg, PA, USA.

Stephen E Robertson, Steve Walker, Susan Jones, et al. 1995. Okapi at TREC-3.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1988. Learning representations by back-propagating errors. *Cognitive modeling*.

Amanda Stent and Srinivas Bangalore. 2014. *Natural Language Generation in Interactive Systems*. Cambridge University Press.

Ilya Sutskever, Oriol Vinyals, and Quoc V. V Le. 2014. Sequence to sequence learning with neural networks. *In Proc. of NIPS*.

Marilyn A. Walker, Rashmi Prasad, and Amanda Stent. 2003. In *Proc. of Interspeech*.

Steve Young, Milica Gašić, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. 2010. The hidden information state model: A practical framework for pomdp-based spoken dialogue management. *Comput. Speech Lang.*, 24(2):150–174.

Steve Young. 2002. Talking to machines (statistically speaking). In *Proc. of Interspeech*.