Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić,
Pei-Hao Su, David Vandyke and **Steve Young**
Cambridge University Engineering Department,
Trumpington Street, Cambridge, CB2 1PZ, UK
{thw28,mg436,nm480,phs26,djv27,sjy}@cam.ac.uk

Abstract

Natural language generation (NLG) is a critical component of spoken dialogue and it has a significant impact both on usability and perceived quality. Most NLG systems in common use employ rules and heuristics and tend to generate rigid and stylised responses without the natural variation of human language. They are also not easily scaled to systems covering multiple domains and languages. This paper presents a statistical language generator based on a semantically controlled Long Short-term Memory (LSTM) structure. The LSTM generator can learn from unaligned data by jointly optimising sentence planning and surface realisation using a simple cross entropy training criterion, and language variation can be easily achieved by sampling from output candidates. With fewer heuristics, an objective evaluation in two differing test domains showed the proposed method improved performance compared to previous methods. Human judges scored the LSTM system higher on informativeness and naturalness and overall preferred it to the other systems.

1 Introduction

The natural language generation (NLG) component provides much of the persona of a spoken dialogue system (SDS), and it has a significant impact on a user's impression of the system. As noted in Stent et al. (2005), a good generator usually depends on several factors: adequacy, fluency, readability, and variation. Previous approaches attacked the NLG problem in different ways. The most common and widely adopted today is the *rule-based* (or *template-based*) approach (Cheyer and Guzzoni, 2007; Mirkovic and Cavedon, 2011). Despite its robustness and adequacy, the frequent repetition of identical, rather stilted, output forms make talking to a *rule-based* generator rather tedious. Furthermore, the approach does not easily scale to large open domain systems(Young et al., 2013; Gašić et al., 2014; Henderson et al., 2014). Hence approaches to NLG are required that can be readily scaled whilst meeting the above requirements.

The *trainable generator* approach exemplified by the HALOGEN (Langkilde and Knight, 1998) and SPaRKy system (Stent et al., 2004) provides a possible way forward. These systems include specific trainable modules within the generation framework to allow the model to adapt to different domains (Walker et al., 2007), or reproduce certain style (Mairesse and Walker, 2011). However, these approaches still require a handcrafted generator to define the decision space within which statistics can be used for optimisation. The resulting utterances are therefore constrained by the predefined syntax and any domain-specific colloquial responses must be added manually.

More recently, *corpus-based* methods (Oh and Rudnicky, 2000; Mairesse and Young, 2014; Wen et al., 2015) have received attention as access to data becomes increasingly available. By defining a flexible learning structure, *corpus-based* methods aim to learn generation directly from data by adopting an over-generation and reranking paradigm (Oh and Rudnicky, 2000), in which final responses are obtained by reranking a set of candidates generated from a stochastic generator. Learning from data directly enables the system to mimic human responses more naturally, removes the dependency on predefined rules, and makes the system easier to build and extend to other domains. As detailed in Sections 2 and 3, however, these existing approaches have weaknesses in the areas of training data efficiency, accuracy and naturalness.

This paper presents a statistical NLG based on a semantically controlled Long Short-term Memory (LSTM) recurrent network. It can learn from unaligned data by jointly optimising its sentence planning and surface realisation components using a simple cross entropy training criterion without any heuristics, and good quality language variation is obtained simply by randomly sampling the network outputs. We start in Section 3 by defining the framework of the proposed neural language generator. We introduce the semantically controlled LSTM (SC-LSTM) cell in Section 3.1, then we discuss how to extend it to a deep structure in Section 3.2. As suggested in Wen et al. (2015), a backward reranker is introduced in Section 3.3 to improve fluency. Training and decoding details are described in Section 3.4 and 3.5.

Section 4 presents an evaluation of the proposed approach in the context of an application providing information about venues in the San Francisco area. In Section 4.2, we first show that our generator outperforms several baselines using objective metrics. We experimented on two different ontologies to show not only that good performance can be achieved across domains, but how easy and quick the development lifecycle is. In order to assess the subjective performance of our system, a quality test and a pairwise preference test are presented in Section 4.3. The results show that our approach can produce high quality utterances that are considered to be more natural and are preferred to previous approaches. We conclude with a brief summary and future work in Section 5.

2 Related Work

Conventional approaches to NLG typically divide the task into sentence planning and surface realisation. Sentence planning maps input semantic symbols into an intermediary form representing the utterance, e.g. a tree-like or template structure, then surface realisation converts the intermediate structure into the final text (Walker et al., 2002; Stent et al., 2004). Although statistical sentence planning has been explored previously, for example, generating the most likely context-free derivations given a corpus (Belz, 2008) or maximising the expected reward using reinforcement learning (Rieser and Lemon, 2010), these methods still rely on a pre-existing, handcrafted generator. To minimise handcrafting, Stent and Molina (2009) proposed learning sentence planning rules directly from a corpus of utterances labelled with Rhetorical Structure Theory (RST) discourse relations (Mann and Thompson, 1988). However, the required corpus labelling is expensive and additional handcrafting is still needed to map the sentence plan to a valid syntactic form.

As noted above, *corpus-based* NLG aims at learning generation decisions from data with minimal dependence on rules and heuristics. A pioneer in this direction is the class-based n-gram language model (LM) approach proposed by Oh and Rudnicky (2000). Ratnaparkhi (2002) later addressed some of the limitations of class-based LMs in the over-generation phase by using a modified generator based on a syntactic dependency tree. Mairesse and Young (2014) proposed a phrase-based NLG system based on factored LMs that can learn from a semantically aligned corpus. Although active learning (Mairesse et al., 2010) was also proposed to allow learning online directly from users, the requirement for human annotated alignments limits the scalability of the system. Another similar approach casts NLG as a template extraction and matching problem, e.g., Angeli et al. (2010) train a set of log-linear models to make a series of generation decisions to choose the most suitable template for realisation. Kondadadi et al. (2013) later show that the outputs can be further improved by an SVM reranker making them comparable to human-authored texts. However, template matching approaches do not generalise well to unseen combinations of semantic elements.

The use of neural network-based (NN) approaches to NLG is relatively unexplored. The stock reporter system ANA by Kukich (1987) is perhaps the first NN-based generator, although generation was only done at the phrase level. Recent advances in recurrent neural network-based language models (RNNLM) (Mikolov et al., 2010; Mikolov et al., 2011a) have demonstrated the value of distributed representations and the ability to model arbitrarily long dependencies. Sutskever et al. (2011) describes a simple variant of the RNN that can generate meaningful sentences by learning from a character-level corpus. More recently, Karpathy and Fei-Fei (2014) have demonstrated that an RNNLM is capable of generating image descriptions by conditioning the network model on a pre-trained convolutional image feature representation. Zhang and Lapata (2014) also describes interesting work using RNNs to generate

Chinese poetry. A forerunner of the system presented here is described in Wen et al. (2015), in which a forward RNN generator, a CNN reranker, and a backward RNN reranker are trained jointly to generate utterances. Although the system was easy to train and extend to other domains, a heuristic gate control was needed to ensure that all of the attribute-value information in the system's response was accurately captured by the generated utterance. Furthermore, the handling of unusual slot-value pairs by the CNN reranker was rather arbitrary. In contrast, the LSTM-based system described in this paper can deal with these problems automatically by learning the control of gates and surface realisation jointly.

Training an RNN with long range dependencies is difficult because of the vanishing gradient problem (Bengio et al., 1994). Hochreiter and Schmidhuber (1997) mitigated this problem by replacing the sigmoid activation in the RNN recurrent connection with a self-recurrent memory block and a set of multiplication gates to mimic the read, write, and reset operations in digital computers. The resulting architecture is dubbed the Long Short-term Memory (LSTM) network. It has been shown to be effective in a variety of tasks, such as speech recognition (Graves et al., 2013b), handwriting recognition (Graves et al., 2009), spoken language understanding (Yao et al., 2014), and machine translation (Sutskever et al., 2014). Recent work by Graves et al. (2014) has demonstrated that an NN structure augmented with a carefully designed memory block and differentiable read/write operations can learn to mimic computer programs. Moreover, the ability to train deep networks provides a more sophisticated way of exploiting relations between labels and features, therefore making the prediction more accurate (Hinton et al., 2012). By extending an LSTM network to be both deep in space and time, Graves (2013) shows the resulting network can used to synthesise handwriting indistinguishable from that of a human.

3 The Neural Language Generator

The generation model proposed in this paper is based on a recurrent NN architecture (Mikolov et al., 2010) in which a 1-hot encoding $\mathbf{w}_t$ of a token[1] w_t is input at each time step t conditioned on a recurrent hidden layer $\mathbf{h}_t$ and outputs the probability distribution of the next token w_{t+1}. Therefore, by sampling input tokens one by one from the output distribution of the RNN until a stop sign is generated (Karpathy and Fei-Fei, 2014) or some constraint is satisfied (Zhang and Lapata, 2014), the network can produce a sequence of tokens which can be lexicalised [2] to form the required utterance.

[1]We use *token* instead of *word* because our model operates on text for which slot values are replaced by its corresponding slot tokens. We call this procedure delexicalisation.

3.1 Semantic Controlled LSTM cell

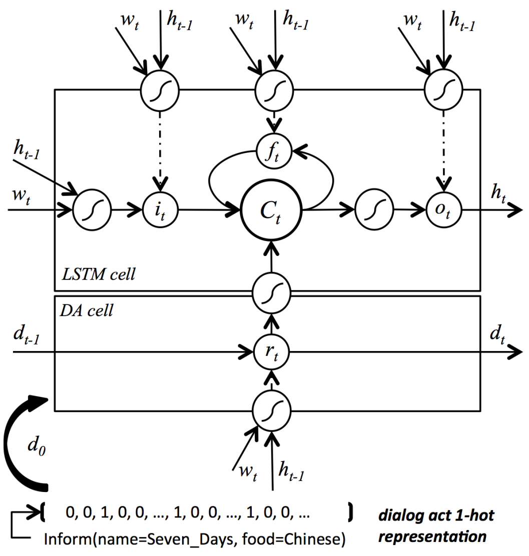


Figure 1: Semantic Controlled LSTM cell proposed in this paper. The upper part is a traditional LSTM cell in charge of surface realisation, while the lower part is a sentence planning cell based on a sigmoid control gate and a dialogue act (DA).

Long Short-term Memory (Hochreiter and Schmidhuber, 1997) is a recurrent NN architecture which uses a vector of memory cells $\mathbf{c}_t \in \mathbb{R}^n$ and a set of elementwise multiplication gates to control how information is stored, forgotten, and exploited inside the network. Of the various different connectivity designs for an LSTM cell (Graves, 2013; Zaremba et al., 2014), the architecture used in this paper is illustrated in Figure 3.1 and defined by the following equations,,

$$\mathbf{i}_t = \sigma(\mathbf{W}_{wi}\mathbf{w}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1}) \quad (1)$$
$$\mathbf{f}_t = \sigma(\mathbf{W}_{wf}\mathbf{w}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1}) \quad (2)$$
$$\mathbf{o}_t = \sigma(\mathbf{W}_{wo}\mathbf{w}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1}) \quad (3)$$
$$\hat{\mathbf{c}}_t = tanh(\mathbf{W}_{wc}\mathbf{w}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1}) \quad (4)$$
$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \hat{\mathbf{c}}_t \quad (5)$$
$$\mathbf{h}_t = \mathbf{o}_t \odot tanh(\mathbf{c}_t) \quad (6)$$

[2]The process of replacing slot token by its value.

where σ is the sigmoid function, $\mathbf{i}_t, \mathbf{f}_t, \mathbf{o}_t \in [0,1]^n$ are input, forget, and output gates respectively, and $\hat{\mathbf{c}}_t$ and $\mathbf{c}_t$ are proposed cell value and true cell value at time t. Note that each of these vectors has a dimension equal to the hidden layer $\mathbf{h}$.

In order to ensure that the generated utterance represents the intended meaning, the generator is further conditioned on a control vector $\mathbf{d}$, a 1-hot representation of the dialogue act (DA) type and its slot-value pairs. Although a related work (Karpathy and Fei-Fei, 2014) has suggested that reapplying this auxiliary information to the RNN at every time step can increase performance by mitigating the *vanishing gradient problem* (Mikolov and Zweig, 2012; Bengio et al., 1994), we have found that such a model also omits and duplicates slot information in the surface realisation. In Wen et al. (2015) simple heuristics are used to turn off slot feature values in the control vector $\mathbf{d}$ once the corresponding slot token has been generated. However, these heuristics can only handle cases where slot-value pairs can be identified by exact matching between the delexicalised surface text and the slot value pair encoded in $\mathbf{d}$. Cases such as binary slots and slots that take don't care values cannot be explicitly delexicalised in this way and these cases frequently result in generation errors.

To address this problem, an additional control cell is introduced into the LSTM to gate the DA as shown in Figure 1. This cell plays the role of *sentence planning* since it manipulates the DA features during the generation process in order to produce a surface realisation which accurately encodes the input information. We call the resulting architecture Semantically Controlled LSTM (SC-LSTM). Starting from the original DA 1-hot vector $\mathbf{d}_0$, at each time step the DA cell decides what information should be retained for future time steps and discards the others,

$$\mathbf{r}_t = \sigma(\mathbf{W}_{wr}\mathbf{w}_t + \alpha\mathbf{W}_{hr}\mathbf{h}_{t-1}) \tag{7}$$

$$\mathbf{d}_t = \mathbf{r}_t \odot \mathbf{d}_{t-1} \tag{8}$$

where $\mathbf{r}_t \in [0,1]^d$ is called the reading gate, and α is a constant. Here $\mathbf{W}_{wr}$ and $\mathbf{W}_{hr}$ act like keyword and key phrase detectors that learn to associate certain patterns of generated tokens with certain slots. Figure 3 gives an example of how these detectors work in affecting DA features inside the network. Equation 5 is then modified so that the cell value $\mathbf{c}_t$ also depends on the DA,

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \hat{\mathbf{c}}_t + tanh(\mathbf{W}_{dc}\mathbf{d}_t) \tag{9}$$

After updating Equation 6 by Equation 9, the output distribution is formed by applying a softmax function g, and the distribution is sampled to obtain the next token,

$$P(w_{t+1}|w_t, w_{t-1}, ...w_0, \mathbf{d}_t) = g(\mathbf{W}_{ho}\mathbf{h}_t) \tag{10}$$

$$w_{t+1} \sim P(w_{t+1}|w_t, w_{t-1}, ...w_0, \mathbf{d}_t). \tag{11}$$

3.2 The Deep Structure

Deep Neural Networks (DNN) enable increased discrimination by learning multiple layers of features, and represent the state-of-the-art for many applications such as speech recognition (Graves et al., 2013b) and natural language processing (Collobert and Weston, 2008). The neural language generator proposed in this paper can be easily extended to be deep in both space and time by stacking multiple LSTM cells on top of the original structure. As shown in Figure 2, *skip connections* are applied to the inputs of all hidden layers as well as between all hidden layers and the outputs (Graves, 2013). This reduces the number of processing steps between the bottom of the network and the top, and therefore mitigates the vanishing gradient problem (Bengio et al., 1994) in the vertical direction. To allow all hidden layer information to influence the reading gate, Equation 7 is changed to

$$\mathbf{r}_t = \sigma(\mathbf{W}_{wr}\mathbf{w}_t + \sum_l \alpha_l \mathbf{W}^l_{hr}\mathbf{h}^l_{t-1}) \tag{12}$$

where l is the hidden layer index and α_l is a layer-wise constant. Since the network tends to overfit when the structure becomes more complex, the dropout technique (Srivastava et al., 2014) is used to regularise the network. As suggested in (Zaremba et al., 2014), dropout was only applied to the non-recurrent connections, as shown in the Figure 2. It was not applied to word embeddings since pre-trained word vectors were used.

3.3 Backward LSTM reranking

One remaining problem in the structure described so far is that the LSTM generator selects words based only on the preceding history, whereas some sentence forms depend on the backward context. Previously, bidirectional networks (Schuster and

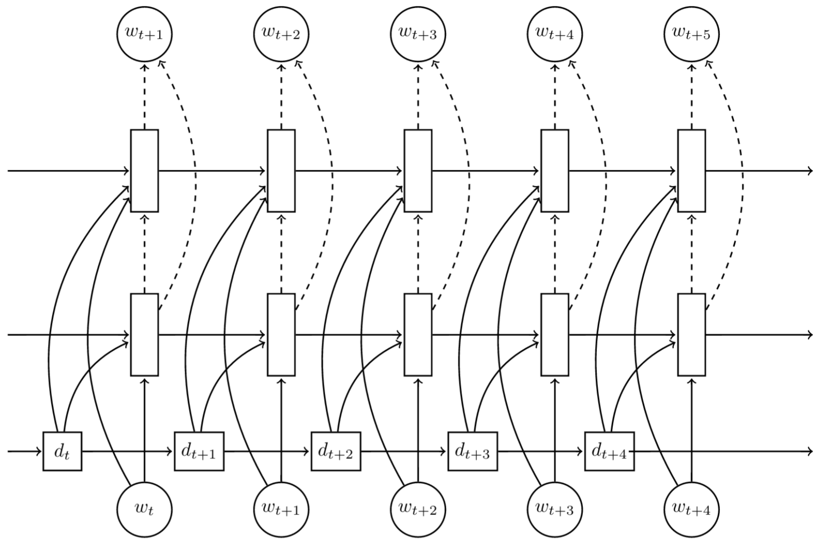


Figure 2: The Deep LSTM generator structure by stacking multiple LSTM layers on top of the DA cell. The skip connection was adopted to mitigate the vanishing gradient, while the dropout was applied on dashed connections to prevent co-adaptation and overfitting.

Paliwal, 1997) have been shown to be effective for sequential problems (Graves et al., 2013a; Sundermeyer et al., 2014). However, applying a bidirectional network directly in the SC-LSTM generator is not straightforward since the generation process is sequential in time. Hence instead of integrating the bidirectional information into one network, we trained another SC-LSTM from backward context to choose best candidates from the forward generator outputs. In our experiments, we also found that by tying the keyword detector weights $\mathbf{W}_{wr}$ (see Equations 7 and 12) of both the forward and backward networks together makes the generator less sensitive to random initialisation.

3.4 Training

The forward generator and the backward reranker were both trained by treating each sentence as a mini-batch. The objective function was the cross entropy error between the predicted word distribution $\mathbf{p}_t$ and the actual word distribution $\mathbf{y}_t$ in the training corpus. An l_2 regularisation term was added to the objective function for every 10 training examples as suggested in Mikolov et al. (2011b). However, further regularisation was required for the reading gate dynamics. This resulted in the following modified cost function for each mini-match (ignoring standard l_2),

$$F(\theta) = \sum_t \mathbf{p}_t^\intercal log(\mathbf{y}_t) + \|\mathbf{d}_T\| + \sum_{t=0}^{T-1} \eta\xi^{\|\mathbf{d}_{t+1}-\mathbf{d}_t\|} \tag{13}$$

where $\mathbf{d}_T$ is the DA vector at the last word index T, and η and ξ are constants set to 10^{-4} and 100, respectively. The second term is used to penalise generated utterances that failed to render all the required slots, while the third term discourages the network from turning more than one gate off in a single time step. The forward and backward networks were structured to share the same set of word embeddings, initialised with pre-trained word vectors (Pennington et al., 2014). The hidden layer size was set to be 80 for all cases, and deep networks were trained with two hidden layers and a 50% dropout rate. All costs and gradients were computed and stochastic gradient descent was used to optimise the parameters. Both networks were trained with back propagation through time (Werbos, 1990). In order to prevent overfitting, early stopping was implemented using a held-out validation set.

3.5 Decoding

The decoding procedure is split into two phases: (a) over-generation, and (b) reranking. In the over-generation phase, the forward generator conditioned on the given DA, is used to sequentially generate utterances by random sampling of the predicted next word distributions. In the reranking phase, the cost of the backward reranker $F_b(\theta)$ is computed. Together with the cost $F_f(\theta)$ from the forward generator, the reranking score R is computed as:

$$R = -(F_f(\theta) + F_b(\theta) + \lambda \mathrm{ERR}) \quad (14)$$

where λ is a tradeoff constant, and the slot error rate ERR is computed by exact matching the slot tokens in the candidate utterances,

$$\mathrm{ERR} = \frac{p+q}{N} \quad (15)$$

where N is the total number of slots in the DA, and p, q is the number of missing and redundant slots in the given realisation. Note that the ERR reranking criteria cannot handle arbitrary slot-value pairs such as binary slots or slots that take the don't care value because they cannot be delexicalised and exactly matched. λ is set to a large value in order to severely penalise nonsensical outputs.

4 Experiments

4.1 Experimental Setup

The target application for our generation system is a spoken dialogue system providing information about certain venues in San Francisco. In order to demonstrate the scalability of the proposed method and its performance in different domains, we tested on two domains that talk about restaurants and hotels respectively. There are 8 system dialogue act types such as *inform* to present information about restaurants, *confirm* to check that a slot value has been recognised correctly, and *reject* to advise that the user's constraints cannot be met. Each domain contains 12 attributes (slots), some are common to both domains and the others are domain specific. The detailed ontologies for the two domains are provided in Table 1. To form a training corpus for each domain, dialogues collected from a previous user trial (Gašić et al., 2015) of a statistical dialogue manager were randomly sampled and shown to workers recruited via the Amazon Mechanical Turk (AMT) service. Workers were shown each dialogue turn by turn and asked to enter an appropriate system response in natural English corresponding to each system DA. For each domain around 5K system utterances were collected from about 1K randomly sampled dialogues. Each categorical value was replaced by a token representing its slot, and slots that appeared multiple times in a DA were merged into one. After processing and grouping each utterance according to its delexicalised DA, we obtained 248 distinct DAs in the restaurant domain and 164 in the hotel domain. The average number of slots per DA for each domain is 2.25 and 1.95, respectively.

Table 1: Ontologies used in the experiments.

	SF Restaurant	SF Hotel
act type	inform, inform_only, reject, confirm, select, request, reqmore, goodbye	
shared	name, type, *pricerange, price, phone, address, postcode, *area, *near	
specific	*food *goodformeal ***kids-allowed**	***hasinternet** ***acceptscards** ***dogs-allowed**

bold=binary slots, *=slots can take "don't care" value

The system was implemented using the Theano library (Bergstra et al., 2010; Bastien et al., 2012), and trained by partitioning each of the collected corpus into a training, validation, and testing set in the ratio 3:1:1. The frequency of each action type and slot-value pair differs quite markedly across the corpus, hence up-sampling was used to make the corpus more uniform. Since our generator works stochastically and the trained networks can differ depending on the initialisation, all the results shown below[3] were averaged over 5 randomly initialised networks. For each DA, we overgenerated 20 utterances and selected the top 5 realisations after reranking. The BLEU-4 metric was used for the objective evaluation (Papineni et al., 2002). Multiple references for each test DA were obtained by mapping them back to the distinct set of DAs, grouping those delexicalised surface forms that have the same DA specification, and then lexicalising those surface forms back to utterances. In addition, the slot error rate (ERR) as described in Section 3.5 was computed as an auxiliary metric alongside the BLEU score. However, for the experiments it is computed at the corpus level, by averaging slot errors over each of the top 5 realisations in the entire corpus. The trade-off weights α between keyword and key phrase detectors as mentioned in Section 3.1 and 3.2 were set to 0.5.

4.2 Objective Evaluation

We compared the single layer semantically controlled LSTM (*sc-lstm*) and a deep version with

[3]Except human evaluation, in which only one set of networks was used.

Table 2: Objective evaluation of the top 5 realisations. Except for handcrafted (*hdc*) and k-nearest neighbour (*kNN*) baselines, all the other approaches ranked their realisations from 20 over-generated candidates.

Method	SF Restaurant		SF Hotel	
	BLEU	ERR(%)	BLEU	ERR(%)
hdc	0.451	0.0	0.560	0.0
kNN	0.602	0.87	0.676	1.87
classlm	0.627	8.70	0.734	5.35
rnn w/o	0.706	4.15	0.813	3.14
lstm w/o	0.714	1.79	0.817	1.93
rnn w/	0.710	1.52	0.815	1.74
lstm w/	0.717	0.63	0.818	1.53
sc-lstm	0.711	0.62	0.802	0.78
+deep	**0.731**	**0.46**	**0.832**	**0.41**

two hidden layers (*+deep*) against several baselines: the handcrafted generator (*hdc*), k-nearest neighbour (*kNN*), class-based LMs (*classlm*) as proposed in Oh and Rudnicky (2000), the heuristic gated RNN as described in Wen et al. (2015) and a similar LSTM variant (*rnn w/* & *lstm w/*), and the same RNN/LSTM but without gates (*rnn w/o* & *lstm w/o*). The handcrafted generator was developed over a long period of time and is the standard generator used for trialling end-to-end dialogue systems (for example (Gašić et al., 2014)). The kNN was implemented by computing the similarity of the test DA 1-hot vector against all of the training DA 1-hot vectors, selecting the nearest and then lexicalising to generate the final surface form. The objective results are shown in Table 2. As can be seen, none of the baseline systems shown in the first block (*hdc*, *kNN*, & *classlm*) are comparable to the systems described in this paper (*sc-lstm* & *+deep*) if both metrics are considered. Setting aside the difficulty of scaling to large domains, the handcrafted generator's (*hdc*) use of predefined rules yields a fixed set of sentence plans, which can differ markedly from the real colloquial human responses collected from AMT, while the class LM approach suffers from inaccurate rendering of information. Although the *kNN* method provides reasonable adequacy i.e. low ERR, the BLEU is low, probably because of the errors in the collected corpus which *kNN* cannot handle but statistical approaches such as LMs can by suppressing unlikely outputs.

The last three blocks in Table 2 compares the proposed method with previous RNN approaches.

Table 3: Real user trial for utterance quality assessment on two metrics (rating out of 3), averaging over top 5 realisations. Statistical significance was computed using a two-tailed Student's t-test, between deep and all others.

Method	**Informativeness**	**Naturalness**
+deep	2.58	**2.51**
sc-lstm	**2.59**	2.50
rnn w/	2.53	2.42*
classlm	2.46**	2.45

$^{*}p < 0.05$ $^{**}p < 0.005$

Table 4: Pairwise preference test among four systems. Statistical significance was computed using two-tailed binomial test.

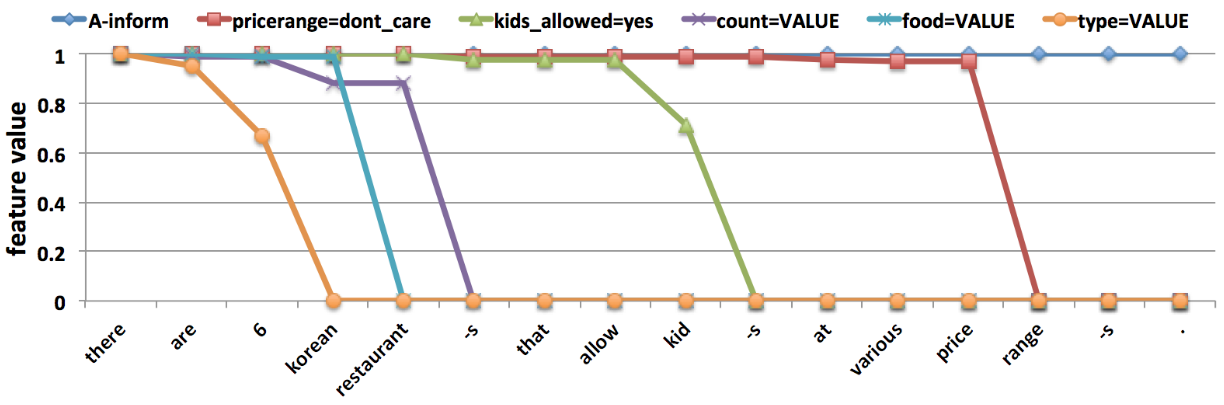
Pref.%	**classlm**	**rnn w/**	**sc-lstm**	**+deep**
classlm	-	46.0	40.9**	37.7**
rnn w/	54.0	-	43.0	35.7*
sc-lstm	59.1*	57	-	47.6
+deep	62.3**	64.3**	52.4	-

$^{*}p < 0.05$ $^{**}p < 0.005$

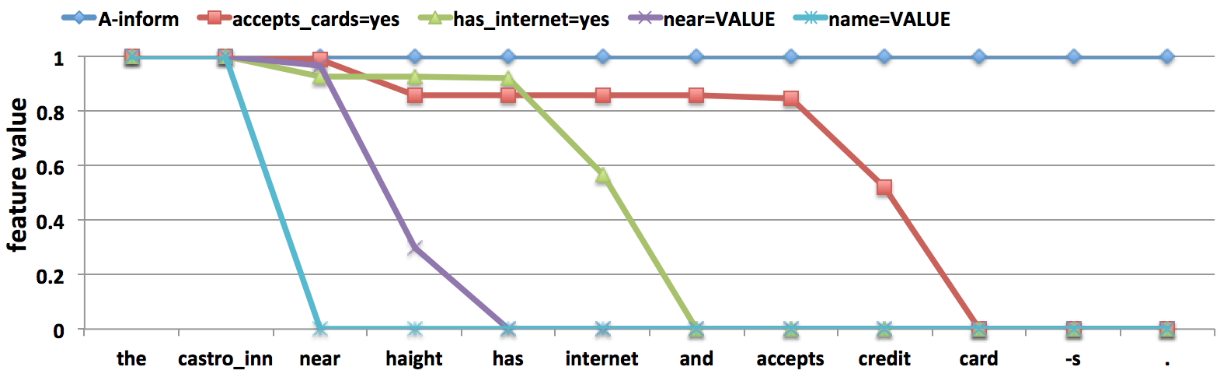
LSTM generally works better than vanilla RNN due to its ability to model long range dependencies more efficiently. We also found that by using gates, whether learned or heuristic, gave much lower slot error rates. As an aside, the ability of the SC-LSTM to learn gates is also exemplified in Figure 3. Finally, by combining the learned gate approach with the deep architecture (*+deep*), we obtained the best overall performance.

4.3 Human Evaluation

Since automatic metrics may not consistently agree with human perception (Stent et al., 2005), human testing is needed to assess subjective quality. To do this, a set of judges were recruited using AMT. For each task, two systems among the four (*classlm*, *rnn w/*, *sc-lstm*, and *+deep*) were randomly selected to generate utterances from a set of newly sampled dialogues in the restaurant domain. In order to evaluate system performance in the presence of language variation, each system generated 5 different surface realisations for each input DA and the human judges were asked to score each of them in terms of informativeness and naturalness (rating out of 3), and also asked to state a preference between the two. Here *informativeness*



(a) An example realisation from SF restaurant domain



(b) An example realisation from SF hotel domain

Figure 3: Examples showing how the SC-LSTM controls the DA features flowing into the network via its learned semantic gates. Despite errors due to sparse training data for some slots, each gate generally learned to detect words and phrases describing a particular slot-value pair.

is defined as whether the utterance contains all the information specified in the DA, and *naturalness* is defined as whether the utterance could plausibly have been produced by a human. In order to decrease the amount of information presented to the judges, utterances that appeared identically in both systems were filtered out. We tested 1000 DAs in total, and after filtering there were approximately 1300 generated utterances per system.

Table 3 shows the quality assessments which exhibit the same general trend as the objective results. The SC-LSTM systems (*sc-lstm* & *+deep*) outperform the class-based LMs (*classlm*) and the RNN with heuristic gates (*rnn w/*) in both metrics. The deep SC-LSTM system (*+deep*) is significantly better than the class LMs (*classlm*) in terms of *informativeness*, and better than the RNN with heuristic gates (*rnn w/*) in terms of *naturalness*. The preference test results are shown in Table 4. Again, the SC-LSTM systems (*sc-lstm* & *+deep*) were significantly preferred by the judges. Moreover, the judges recorded a strong preference for the deep approach (*+deep*) compared to the others, though the preference is not significant when comparing to its shallow counterpart (*sc-lstm*). Example dialogue acts and their top-5 realisations are shown in Table 5.

5 Conclusion and Future Work

In this paper we have proposed a neural network-based generator that is capable of generating natural linguistically varied responses based on a deep, semantically controlled LSTM architecture which we call SC-LSTM. The generator can be trained on unaligned data by jointly optimising its sentence planning and surface realisation components using a simple cross entropy criterion without any heuristics or handcrafting. We found that the SC-LSTM model achieved the best overall performance on two objective metrics across two different domains. An evaluation by human judges also confirmed that the SC-LSTM approach is strongly preferred to a variety of existing methods.

This work represents a line of research that tries

Table 5: Samples of top 5 realisations from the deep SC-LSTM (+deep) system output.

#	**Example Dialogue Acts and Realizations from SF Restaurant Domain**
1	inform(name="red door cafe", goodformeal="breakfast", area="cathedral hill", kidsallowed="no") *red door cafe is a good restaurant for breakfast in the area of cathedral hill and does not allow children .* *red door cafe is a good restaurant for breakfast in the cathedral hill area and does not allow children .* *red door cafe is a good restaurant for breakfast in the cathedral hill area and does not allow kids .* *red door cafe is good for breakfast and is in the area of cathedral hill and does not allow children .* *red door cafe does not allow kids and is in the cathedral hill area and is good for breakfast .*
2	informonly(name="dosa on fillmore and kiss seafood", pricerange="expensive", near="lower pacific heights") *there is no place other than dosa on fillmore and kiss seafood that are expensive near to lower pacific heights .* *dosa on fillmore and kiss seafood is the only expensive restaurant near lower pacific heights .* *the only listed restaurant near lower pacific heights in the expensive price range is dosa on fillmore and kiss seafood .* *i apologize , dosa on fillmore and kiss seafood is the only expensive restaurant near lower pacific heights .* *i apologize , dosa on fillmore and kiss seafood are the only expensive restaurants near lower pacific heights .*
#	**Example Dialogue Acts and Realizations from SF Hotel Domain**
3	inform(type="hotel",count="182",dogsallowed="dontcare") *there are 182 hotels if you do not care whether dogs are allowed .* *there are 182 hotels if you do not care whether they allow dogs .* *182 hotels are available if dogs allowed or not is not an issue .* *there are 182 hotels if allowing dogs or not is not an issue .* *there are 182 hotels if whether dogs are allowed does not matter .*
4	informonly(name="red victorian bed breakfast",acceptscreditcards="yes",near="haight",hasinternet="yes") *red victorian bed breakfast is the only hotel near haight and accepts credit cards and has internet .* *red victorian bed breakfast is the only hotel near haight and has internet and accepts credit cards .* *red victorian bed breakfast is the only hotel near haight that accept credit cards and offers internet .* *the red victorian bed breakfast has internet and near haight , it does accept credit cards .* *the red victorian bed breakfast is the only hotel near haight that accepts credit cards , and offers internet .*

to model the NLG problem in a unified architecture, whereby the entire model is end-to-end trainable from data. We contend that this approach can produce more natural responses which are more similar to colloquial styles found in human conversations. Another key potential advantage of neural network based language processing is the implicit use of distributed representations for words and a single compact parameter encoding of the information to be conveyed. This suggests that it should be possible to further condition the generator on some dialogue features such discourse information or social cues during the conversation. Furthermore, adopting a corpus based regime enables domain scalability and multilingual NLG to be achieved with less cost and a shorter lifecycle. These latter aspects will be the focus of our future work in this area.

6 Acknowledgements

Tsung-Hsien Wen and David Vandyke are supported by Toshiba Research Europe Ltd, Cambridge Research Laboratory.

References

Gabor Angeli, Percy Liang, and Dan Klein. 2010. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on EMNLP*, EMNLP '10. Association for Computational Linguistics.

Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. 2012. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop.

Anja Belz. 2008. Automatic generation of weather forecast texts using comprehensive probabilistic generation-space models. *Natural Language Engineering*.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. 2010. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference*.

Adam Cheyer and Didier Guzzoni. 2007. Method and apparatus for building an intelligent automated assistant. US Patent App. 11/518,292.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*.

Milica Gašić, Dongho Kim, Pirros Tsiakoulis, Catherine Breslin, Matthew Henderson, Martin Szummer, Blaise Thomson, and Steve Young. 2014. Incremental on-line adaptation of pomdp-based dialogue managers to extended domains. In *In Proceedings on InterSpeech*.

Milica Gašić, Dongho Kim, Pirros Tsiakoulis, and Steve Young. 2015. Distributed dialogue policies for multi-domain statistical dialogue management. In *In Proceedings on ICASSP*.

Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. 2009. A novel connectionist system for unconstrained handwriting recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013a. Speech recognition with deep recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. 2013b. Speech recognition with deep recurrent neural networks. *CoRR*, abs/1303.5778.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *CoRR*, abs/1410.5401.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014. Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In *Proceedings of IEEE Spoken Language Technology*.

Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*.

Andrej Karpathy and Li Fei-Fei. 2014. Deep visual-semantic alignments for generating image descriptions. *CoRR*.

Ravi Kondadadi, Blake Howald, and Frank Schilder. 2013. A statistical nlg framework for aggregated planning and realization. In *Proceedings of the 51st Annual Meeting of the ACL*. Association for Computational Linguistics.

Karen Kukich. 1987. Where do phrases come from: Some preliminary experiments in connectionist phrase generation. In *Natural Language Generation*. Springer Netherlands.

Irene Langkilde and Kevin Knight. 1998. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the 36th Annual Meeting of the ACL*, ACL '98.

François Mairesse and Marilyn A. Walker. 2011. Controlling user perceptions of linguistic style: Trainable generation of personality traits. *Computer Linguistics*.

François Mairesse and Steve Young. 2014. Stochastic language generation in dialogue using factored language models. *Computer Linguistics*.

François Mairesse, Milica Gašić, Filip Jurčíček, Simon Keizer, Blaise Thomson, Kai Yu, and Steve Young. 2010. Phrase-based statistical language generation using graphical models and active learning. In *Proceedings of the 48th ACL*, ACL '10.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*.

Tomáš Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. In *In Proceedings on IEEE SLT workshop*.

Tomáš Mikolov, Martin Karafit, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *In Proceedings on InterSpeech*.

Tomáš Mikolov, Stefan Kombrink, Lukáš Burget, Jan H. Černocký, and Sanjeev Khudanpur. 2011a. Extensions of recurrent neural network language model. In *ICASSP, 2011 IEEE International Conference on*.

Tomáš Mikolov, Stefan Kombrink, Anoop Deoras, Lukáš Burget, and Jan Černocký. 2011b. Rnnlm - recurrent neural network language modeling toolkit. In *In Proceedings on ASRU*.

Danilo Mirkovic and Lawrence Cavedon. 2011. Dialogue management using scripts. EP Patent 1,891,625.

Alice H. Oh and Alexander I. Rudnicky. 2000. Stochastic language generation for spoken dialogue systems. In *Proceedings of the 2000 ANLP/NAACL Workshop on Conversational Systems - Volume 3*, ANLP/NAACL-ConvSyst '00.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on ACL*. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on EMNLP*. Association for Computational Linguistics.

Adwait Ratnaparkhi. 2002. Trainable approaches to surface natural language generation and their application to conversational dialog systems. *Computer Speech and Language*.

Verena Rieser and Oliver Lemon. 2010. Natural language generation as planning under uncertainty for spoken dialogue systems. In *Empirical Methods in Natural Language Generation*. Springer-Verlag.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*.

Amanda Stent and Martin Molina. 2009. Evaluating automatic extraction of rules for sentence plan construction. In *Proceedings of SIGdial*. Association for Computational Linguistics.

Amanda Stent, Rashmi Prasad, and Marilyn Walker. 2004. Trainable sentence planning for complex information presentation in spoken dialog systems. In *In Proceedings of the Annual Meeting of the ACL*.

Amanda Stent, Matthew Marge, and Mohit Singhai. 2005. Evaluating evaluation methods for generation in the presence of variation. In *in Proceedings of CICLing 2005*.

Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. 2014. Translation modeling with bidirectional recurrent neural networks. In *Proceedings of the 2014 Conference on EMNLP*. Association for Computational Linguistics.

Ilya Sutskever, James Martens, and Geoffrey E. Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*. ACM.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *CoRR*.

Marilyn A Walker, Owen C Rambow, and Monica Rogati. 2002. Training a sentence planner for spoken dialogue using boosting. *Computer Speech and Language*.

Marilyn Walker, Amanda Stent, Franois Mairesse, and Rashmi Prasad. 2007. Individual and domain adaptation in sentence planning for dialogue. *Journal of Artificial Intelligence Research (JAIR*.

Tsung-Hsien Wen, Milica Gašić, Dongho Kim, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. In *Proceedings of SIGdial*. Association for Computational Linguistics.

Paul J Werbos. 1990. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*.

Kaisheng Yao, Baolin Peng, Yu Zhang, Dong Yu, Geoffrey Zweig, and Yangyang Shi. 2014. Spoken language understanding using long short-term memory neural networks. In *In Proceedings on IEEE SLT workshop*. IEEE Institute of Electrical and Electronics Engineers.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D. Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *CoRR*, abs/1409.2329.

Xingxing Zhang and Mirella Lapata. 2014. Chinese poetry generation with recurrent neural networks. In *Proceedings of the 2014 Conference on EMNLP*. Association for Computational Linguistics, October.