Multiresolution Recurrent Neural Networks: An Application to Dialogue Response Generation

Iulian Vlad Serban*
University of Montreal
2920 chemin de la Tour,
Montréal, QC, Canada

Tim Klinger and **Gerald Tesauro**
IBM Research
T. J. Watson Research Center,
Yorktown Heights, NY, USA

Kartik Talamadupula and **Bowen Zhou**
IBM Research
T. J. Watson Research Center,
Yorktown Heights, NY, USA

Yoshua Bengio† and **Aaron Courville**
University of Montreal
2920 chemin de la Tour,
Montréal, QC, Canada

Abstract

We introduce a new class of models called *multiresolution recurrent neural networks*, which explicitly model natural language generation at multiple levels of abstraction. The models extend the sequence-to-sequence framework to generate two parallel stochastic processes: a sequence of high-level coarse tokens, and a sequence of natural language words (e.g. sentences). The coarse sequences follow a latent stochastic process with a factorial representation, which helps the models generalize to new examples. The coarse sequences can also incorporate task-specific knowledge, when available. In our experiments, the coarse sequences are extracted using automatic procedures, which are designed to capture compositional structure and semantics. These procedures enable training the multiresolution recurrent neural networks by maximizing the exact joint log-likelihood over both sequences. We apply the models to dialogue response generation in the technical support domain and compare them with several competing models. The multiresolution recurrent neural networks outperform competing models by a substantial margin, achieving state-of-the-art results according to both a human evaluation study and automatic evaluation metrics. Furthermore, experiments show the proposed models generate more fluent, relevant and goal-oriented responses.

Introduction

Recurrent neural networks (RNNs) have recently shown excellent performance on tasks such as language modelling (Graves 2012; Mikolov and others 2010), machine translation (Sutskever, Vinyals, and Le 2014; Cho and others 2014) and speech recognition (Hinton and others 2012). Inspired by these advances, researchers have started to investigate such models for dialogue applications (Ritter, Cherry, and Dolan 2011; Lowe et al. 2015; Sordoni et al. 2015b; Shang, Lu, and Li 2015; Li et al. 2016; Serban et al. 2016). In particular, researchers have proposed several sequence-to-sequence (Seq2Seq) encoder-decoder models for dialogue response generation. In this task, the model must generate an appropriate response given a dialogue context. Although this framework differs from the well-established goal-oriented setting (Gorin, Riccardi, and Wright 1997; Young 2000; Singh et al. 2002), these models have been applied to several real-world applications. One example is Microsoft's system Xiaolic, which now interacts with millions of users every day (Markoff and Mozur 2015). Another example is Google's Smart Reply system (Kannan et al. 2016). However, in spite of recent advances, current models cannot effectively take dialogue context into account and generate meaningful and diverse on-topic responses (Li et al. 2016).

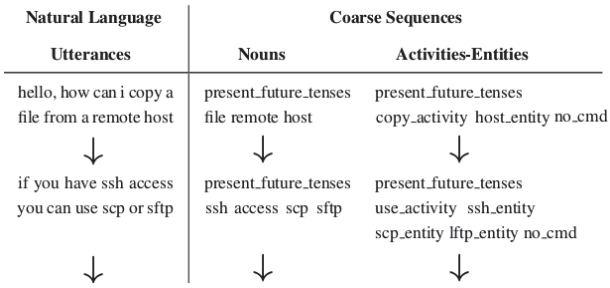

Figure 1: Example dialogue with coarse representations, where arrows indicate turn change in the dialogue.

To overcome these problems, we construct probabilistic models to represent and reason at several levels of abstraction. Explicitly representing multiple levels of abstraction should make it easier for models to remember and reason over long-term context, and to generate appropriate responses with compositional structure. For example, for technical support applications, this should help the models identify the user's problem and generate response with appropriate, and even complex, technical instructions. Specifically, we propose a new class of models – called *multiresolution recurrent neural networks* (MrRNNs) – that generate two parallel stochastic sequences: a sequence of high-level coarse tokens (coarse sequences), and a sequence of low-level natural language words (utterances). The coarse sequences follow a latent stochastic

*I. V. S. was at IBM Research when this work was carried out.
†Y. B. is a CIFAR Senior Fellow.

process with a factorial representation, which helps the models generalize to new examples. The coarse sequences generate the natural language utterances through a hierarchical generation process. This generation process adds high-level, compositional structure to the model, which helps generate meaningful and on-topic responses. Furthermore, the coarse sequences can incorporate task-specific knowledge. In our experiments, the coarse sequences are defined either as noun sequences or activity-entity pairs (predicate-argument pairs) extracted from the natural language utterances (see Figure 1). The extraction allows training MrRNNs using the joint log-likelihood over all observed sequences, in contrast to standard Seq2Seq models which are trained using the (marginal) log-likelihood. We apply MrRNNs to dialogue response generation in the Ubuntu technical support domain, and compare them with recently proposed models. MrRNNs outperform competing approaches by a substantial margin according to both a human evaluation study and automatic evaluation metrics achieving a new state-of-the-art result. Furthermore, the results indicate MrRNNs hold significant potential for goal-oriented dialogue applications.

Model Architecture

Recurrent Neural Network Language Model

We first introduce the well-established recurrent neural network language model (RNNLM) (Mikolov and others 2010; Bengio et al. 2003). Let $w_1, \ldots, w_N$ be a sequence of discrete variables, such as words in a document, called tokens where $w_n \in V$ for set (vocabulary) V^w. The RNNLM is a probabilistic generative model, with parameters θ, which decomposes the probability over tokens:

$$P_\theta(w_1, \ldots, w_N) = \prod_{n=1}^{N} P_\theta(w_n | w_1, \ldots, w_{n-1}). \quad (1)$$

where the output distribution uses a softmax RNN:

$$P_\theta(w_{n+1} = v | w_1, \ldots, w_n) = \frac{\exp(g(h_n, v))}{\sum_{v' \in V} \exp(g(h_n, v'))}, \quad (2)$$

$$h_n = f(h_{n-1}, w_n),\ g(h_n, v) = O_v^{\mathrm{T}} h_n\ \forall v \in V, \quad (3)$$

where f is the hidden state update function. We will assume it is either the LSTM or GRU gating unit (Hochreiter and Schmidhuber 1997; Cho and others 2014).[1] The gating units may also be bidirectional. The matrix $I \in \mathbb{R}^{d_h \times |V|}$ is the input word embedding matrix, where row i contains the embedding for word index i and $d_h \in \mathbb{N}$ is the word embedding dimensionality. Similarly, the matrix $O \in \mathbb{R}^{d_h \times |V|}$ is the output word embedding matrix. According to the model, the probability of observing a token w at position $n+1$ increases if the context vector h_n has a higher dot-product with the word embedding corresponding to token w. The model parameters are usually learned by maximizing the log-likelihood (equivalent to minimizing the cross-entropy) on the training set using gradient descent.

[1] For the LSTM gating unit, we consider the hidden state h_m to be the input cell and memory cell concatenated.

Hierarchical Recurrent Encoder-Decoder

Our work builds upon the hierarchical recurrent encoder-decoder model (HRED) (Sordoni et al. 2015a), which was previously proposed for dialogue (Serban et al. 2016). HRED decomposes a dialogue into a two-level hierarchy: a sequence of utterances, each of which is a sequence of words. Let $\mathbf{w}_1, \ldots, \mathbf{w}_N$ be the sequence of utterances with length N, where $\mathbf{w}_n = (w_{n,1}, \ldots, w_{n,K_n})$ is the n'th utterance consisting of K_n discrete tokens from vocabulary V^w. HRED decomposes the probability distribution as:

$$P_\theta(\mathbf{w}_1, \ldots, \mathbf{w}_N) = \prod_{n=1}^{N} P_\theta(\mathbf{w}_n | \mathbf{w}_{<n}),$$
$$= \prod_{n=1}^{N} \prod_{m=1}^{K_n} P_\theta(w_{n,m} | w_{n,<m}, \mathbf{w}_{<n}). \quad (4)$$

This decomposition allows HRED to capture the hierarchical structure in natural language sequences, such as dialogue. HRED consists of three layers. First, each utterance sequence is encoded into a real-valued vector by an *encoder* RNN:

$$h^{\mathrm{e}}_{n,0} = \mathbf{0},\ \ h^{\mathrm{e}}_{n,m} = f^{\mathrm{e}}_\theta(h^{\mathrm{e}}_{n,m-1}, w_{n,m})\ \forall m = 1, \ldots, K_n,$$

where f^{e}_θ is the *encoder* RNN gating function. The last hidden state, h^{e}_{n,K_n}, summarizes the utterance and is given to the higher-level *context* RNN:

$$h^{\mathrm{c}}_0 = \mathbf{0},\ \ h^{\mathrm{c}}_n = f^{\mathrm{c}}_\theta(h^{\mathrm{c}}_{n-1}, h^{\mathrm{e}}_{n,K_n})$$

where f^{c}_θ is the *context* RNN gating function taking two vectors as input. The *context* RNN acts like a memory module which remembers things over long time scales. As such, the hidden state h^{c}_{n-1} summarizes the dialogue up to the $(n-1)$'th utterance. This is given to the *decoder* RNN:

$$h^{\mathrm{d}}_{n,0} = \mathbf{0},\ h^{\mathrm{d}}_{n,m} = f^{\mathrm{d}}_\theta(h^{\mathrm{d}}_{n,m-1}, h^{\mathrm{c}}_{n-1}, w_{n,m})$$
$$\forall m = 1, \ldots, K_n,$$

where f^{d}_θ is the LSTM gating function for the *decoder* RNN, which takes as input three vectors. The hidden state $h^{\mathrm{d}}_{n,m}$ is transformed through a one-layer neural network yielding the output vector $h^{\mathrm{o}}_{n,m}$. Finally, the output distribution is given by eq. (2) where h_n is replaced by $h^{\mathrm{o}}_{n,m}$.

Multiresolution RNN (MrRNN)

We now introduce the multiresolution recurrent neural network (MrRNN). As before, let $\mathbf{w}_1, \ldots, \mathbf{w}_N$ be the sequence of utterances. Let $\mathbf{z}_1, \ldots, \mathbf{z}_N$ be the corresponding sequence of (high-level) *coarse* constituent sequences, also of length N, where $\mathbf{z}_n = (z_{n,1}, \ldots, z_{n,L_n})$ is the n'th constituent sequence consisting of L_n discrete tokens from vocabulary V^z. For example, each coarse sequence $\mathbf{z}_n$ could represent the nouns of the corresponding utterance $\mathbf{w}_n$.

MrRNN is a probabilistic generative model over the sequences $\mathbf{w}_1, \ldots, \mathbf{w}_N$ and $\mathbf{z}_1, \ldots, \mathbf{z}_N$. MrRNN generates the n'th coarse sequence $\mathbf{z}_n$ by conditioning only on previous coarse sequences $\mathbf{z}_1, \ldots, \mathbf{z}_{n-1}$. Then, MrRNN generates the

n'th utterance $\mathbf{w}_n$ by conditioning on previous utterances $\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}$ and coarse sequences $\mathbf{z}_1, \ldots, \mathbf{z}_n$. Formally:

$$P_\theta(\mathbf{w}_1, \ldots, \mathbf{w}_N, \mathbf{z}_1, \ldots, \mathbf{z}_N) = \prod_{n=1}^{N} P_\theta(\mathbf{z}_n|\mathbf{z}_1, \ldots, \mathbf{z}_{n-1}) \\ P_\theta(\mathbf{w}_n|\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}, \mathbf{z}_1, \ldots, \mathbf{z}_n),$$

where the conditional distributions decompose as:

$$P_\theta(\mathbf{z}_n|\mathbf{z}_1, \ldots, \mathbf{z}_{n-1}) \\ = \prod_{m=1}^{L_n} P_\theta(z_{n,m}|z_{n,1}, \ldots, z_{n,m-1}, \mathbf{z}_1, \ldots, \mathbf{z}_{n-1}) \\ P_\theta(\mathbf{w}_n|\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}, \mathbf{z}_1, \ldots, \mathbf{z}_n) \\ = \prod_{m=1}^{K_n} P_\theta(w_{n,m}|w_{n,1}, \ldots, w_{n,m-1}, \mathbf{w}_{<n}, \mathbf{z}_{<n}, \mathbf{z}_n)$$

The coarse sequences follow a latent stochastic process – similar to hidden Markov models – which does not depend on the utterances. This helps the model generalize to new dialogues, and enables training the model efficiently.

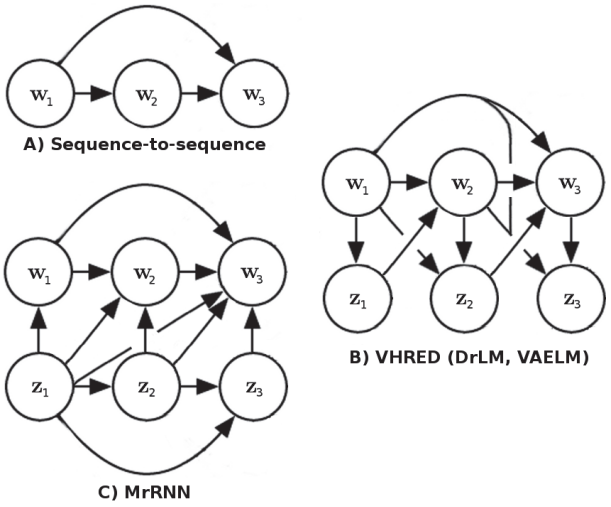

Figure 2: Probabilistic directed graphical models for natural language generation. Variables $\mathbf{w}$ represent natural language utterances. Variables $\mathbf{z}$ represent discrete or continuous stochastic latent variables. (A): Most Seq2Seq models follow a shallow generation process. This is problematic because it forces the model to generate compositional and long-term structure incrementally as it samples each word. (B): VHRED, VAELM and DrLM expand the generation process by adding one latent variable for each utterance. (C): MrRNN expands the generation process by adding a sequence of discrete stochastic variables for each utterance, which follow their own latent stochastic process.

The model over $\mathbf{z}_1, \ldots, \mathbf{z}_N$ is called the *coarse sub-model*, and computes the conditional distribution $P_\theta(\mathbf{z}_n|\mathbf{z}_1, \ldots, \mathbf{z}_{n-1})$ as the standard HRED model. The model over $\mathbf{w}_1, \ldots, \mathbf{w}_N$ is called the *natural language sub-model*. This sub-model computes $P_\theta(\mathbf{w}_n|\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}, \mathbf{z}_1, \ldots, \mathbf{z}_n)$ as the HRED model applied to the sequences $\mathbf{w}_1, \ldots, \mathbf{w}_N$, but conditioned on $\mathbf{z}_1, \ldots, \mathbf{z}_N$. It consists of a (GRU-gated) *coarse prediction encoder* RNN, which encodes the coarse tokens $\mathbf{z}_1, \ldots, \mathbf{z}_n$:

$$h^{\mathrm{p}}_{0,0} = \mathbf{0}, h^{\mathrm{p}}_{n,0} = h^{\mathrm{p}}_{n-1,L_{n-1}} \\ h^{\mathrm{p}}_{n,m} = f^{\mathrm{p}}_\theta(h^{\mathrm{p}}_{n,m-1}, z_{n,m}) \; \forall m = 1, \ldots, L_n,$$

where f^{p}_θ is the gating function. This RNN summarizes all the coarse tokens up to the $(n-1)$'th utterance in the vector $h^{\mathrm{p}}_{n-1,L_{n-1}}$. This summary is different from the *context* RNN in the coarse sub-model, because it is used specifically to generate the next natural language utterance. For this reason, the *coarse prediction encoder* RNN has separate word embedding parameters. The hidden state $h^{\mathrm{p}}_{n-1,L_{n-1}}$ is given as input to the natural language *decoder* RNN – together with the hidden state of the natural language *context* RNN. As before, the output distribution is given by eq. (2).

For training, we assume that both $\mathbf{z}_1, \ldots, \mathbf{z}_N$ and $\mathbf{w}_1, \ldots, \mathbf{w}_N$ are observed and optimize model parameters w.r.t. the joint log-likelihood over both sequences. At test time, we generate model responses by approximating the response with highest probability:

$$\arg\max_{\mathbf{w}_n, \mathbf{z}_n} P_\theta(\mathbf{w}_n, \mathbf{z}_n|\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}, \mathbf{z}_1, \ldots, \mathbf{z}_{n-1}) \\ \approx \arg\max_{\mathbf{w}_n} P_\theta(\mathbf{w}_n|\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}, \mathbf{z}_1, \ldots, \mathbf{z}_{n-1}, \mathbf{z}_n) \\ \arg\max_{\mathbf{z}_n} P_\theta(\mathbf{z}_n|\mathbf{z}_1, \ldots, \mathbf{z}_{n-1}), \quad (5)$$

where we approximate each argmax using beam search.

Relation to Other Generative Models

Figure 2 illustrates how MrRNN is related to other generative models. The standard RNNLM (Mikolov and others 2010) follow a shallow generation process, where each word is generated conditioned on all previous words (Figure 2, A)). The same applies to most Seq2Seq models for dialogue (Sordoni et al. 2015b; Shang, Lu, and Li 2015; Li et al. 2016; Serban et al. 2016) and other models (Bahdanau, Cho, and Bengio 2015; Weston, Chopra, and Bordes 2015; Graves, Wayne, and Danihelka 2014; Kumar et al. 2016). However, this is problematic because it forces the model to generate compositional and long-term structure incrementally as it samples each word (Chung et al. 2015; Serban et al. 2017). For example, if $\mathbf{w}$ and $\mathbf{w}'$ are two utterances with high probability, the model has to choose between sampling either $\mathbf{w}$ or $\mathbf{w}'$ at the first word where they differ.

The VHRED model (Serban et al. 2017; Bowman et al. 2016) expands the shallow generation process by adding a multivariate continuous stochastic latent variable $\mathbf{z}_n$ for each utterance, conditioned on the previous utterances (Figure 2, B)). VHRED generates an utterance by first sampling $\mathbf{z}_n$, and then generates the utterance $\mathbf{w}_n$ word by word. In contrast to MrRNN, VHRED generates only one continuous high-level variable per utterance, while MrRNN generates a sequence of discrete high-level variables per utterance. By operating on sequences of high-level variables, MrRNN can model high-level variables through a factorial representation. This

representation adds compositional structure to the generated utterances and improves generalization to new high-level sequences. The models also differ w.r.t. their dependency assumptions: VHRED conditions the high-level variable on previous utterances, while MrRNN conditions its high-level variables on previous high-level variables. Similar to hidden Markov models, MrRNN assumes the high-level variables follow dynamics independent of the low-level variables. When this assumption is approximately satisfied – which we will argue is the case in the technical support domain – it will help the models to generalize to new examples. Also related is the DrLM model (Ji, Haffari, and Eisenstein 2016), which generates a discrete stochastic variable z_n for each utterance, conditioned on the previous utterance. In contrast to MrRNN, DrLM uses only a single discrete high-level variable per utterance and only conditions this variable on the immediate previous utterance. Other RNNs with stochastic latent variables include the Variational Recurrent Neural Network (Chung et al. 2015), the Variational Recurrent Autoencoder (Fabius and van Amersfoort 2015) and the Structured Variational Autoencoder (SVAE) (Johnson et al. 2016). However, these models have not been applied to natural language generation.

Experiments

We experiment on the task of language response generation for dialogue. Given a dialogue context consisting of one or more utterances, the model must generate the next response in that dialogue. The specific task we consider is technical support for the Ubuntu operating system; the data we use is the Ubuntu Dialogue Corpus developed by Lowe et al (2015). The corpus consists of about half a million natural language dialogues extracted from the *#Ubuntu* Internet Relayed Chat (IRC) channel. Users entering this chat channel usually have a specific technical problem. Typically, users first describe their problem, following which other users try to help them resolve it. The technical problems range from software-related issues (e.g. installing or upgrading existing software) and hardware-related issues (e.g. fixing broken drivers), to informational needs (e.g. finding software).

Coarse Sequence Representations

We experiment with two procedures for extracting the coarse sequences. The first is broadly applicable to a variety of dialogue tasks, and the second is specific to Ubuntu:

Noun Representation This procedure aims to exploit the basic high-level structure of natural language discourse. It is motivated by the observation that dialogues are topic-driven and that these topics may be characterized by nouns. As such, the procedure requires a part-of-speech tagger to identify the nouns in the dialogue. The procedure uses a set of 84 predefined stop words. It maps a natural language utterance to its corresponding coarse representation by extracting all the nouns using the part-of-speech tagger, and then removing all stop words and repeated words (keeping only the first occurrence of a given word). Utterances without nouns are assigned the "no_nouns" token. Finally, the procedure adds the tense of each utterance to the beginning of the coarse sequence.

Activity-Entity Representation This procedure is specific to the Ubuntu technical support task, for which it exploits domain knowledge related to technical problem solving. It is motivated by the observation that most dialogues are centered around *activities* and *entities*. For example, users frequently state a specific problem they want to resolve, e.g. *'how do I install program X?'* or *'My driver X doesn't work, how do I fix it?'* In response to such questions, other users often respond with specific instructions, e.g. *'Go to website X to download software Y'* or *'Try to execute command X'*. In these cases, the principal information resides in the technical entities (e.g. *X*, *Y*) and in the verbs (e.g. *execute*, *download*). In order to capture this information, the procedure uses a set of 192 activities (verbs) created by manual inspection, and a set of 3115 technical entities and 230 frequent terminal commands extracted automatically from available package managers and from the web. The procedure extracts the verbs from each natural language utterance, and maps the natural language to coarse representation verbs from the activity set and entities from the technical entity set. If the utterance does not contain any activity, the "none_activity" token is used. The procedure also appends a binary variable to the end of the coarse representation indicating if a terminal command was detected in the utterance. Finally, the procedure adds the tense of each utterance to the beginning of the coarse sequence.

Both extraction procedures are applied separately for each utterance in a dialogue. Figure 1 gives an example of extracted representations.[2]

Models

We implement all models in Theano (Theano Development Team 2016). We train all models w.r.t. the log-likelihood or joint log-likelihood on the training set using Adam (Kingma and Ba 2015). The models are trained using early stopping with patience based on the validation set log-likelihood. We choose model hyperparameters – such as the number of hidden units, word embedding dimensionality, and learning rate – based on the validation set log-likelihood. We use gradient clipping to stop the parameters from exploding (Pascanu, Mikolov, and Bengio 2012). We define the 20,000 most frequent words as the vocabulary, and map all other words to a special *unknown* token. Based on several experiments, we fix the word embedding dimensionality to size 300 for all models. At test time, we use a beam search of size 5 for generating the model responses.

Baseline Models We compare our models to several competing models proposed previously in the literature. The first baseline is the standard RNNLM with LSTM gating function (Mikolov and others 2010) (LSTM). This model is identical to the Seq2Seq LSTM model (Sutskever, Vinyals, and Le 2014), when the Seq2Seq LSTM model is trained to generate every utterance conditioned on all past utterances. The sec-

[2]The pre-processed Ubuntu Dialogue Corpus and the coarse representations can be downloaded at http://www.iulianserban.com/Files/UbuntuDialogueCorpus.zip and https://github.com/julianser/Ubuntu-Multiresolution-Tools.

ond baseline is HRED with LSTM gating function for the *decoder* RNN; and GRU gating function for the *encoder* RNN and *context* RNN. This model was developed specifically for dialogue response generation (Serban et al. 2016). The RNNLM model has 2000 hidden units with the LSTM gating function. The HRED model has 500, 1000, and 500 hidden units respectively for the *encoder* RNN, *context* RNN, and *decoder* RNN. The third baseline is the latent variable hierarchical recurrent encoder-decoder (VHRED), which extends HRED (Serban et al. 2017). We use the publicly available responses provided by Serban et al. (2017).

In order to investigate the importance of the hierarchical generation process, we introduce a fourth baseline, called *HRED + Activity-Entity Features*. This model simplifies MrRNN, such that the natural language *decoder* RNN is conditioned only on the *past* coarse tokens. In other words, this baseline has access to past activity-entity pairs – encoded using a GRU RNN – but it does not contain a sub-model for generating future coarse sequences. As such, the baseline has access to the exact same information as MrRNN, but does not make use of a hierarchical generation process. We use this baseline to evaluate the importance of the hierarchical generation process. Based on the validation set log-likelihood, we select the model architecture to have 500, 1000, and 2000 hidden units respectively for the *encoder* RNN, *context* RNN, and *decoder* RNN. The GRU RNN, which encodes the past coarse-level activity-entities sequences, has 500 hidden units.

MrRNN The coarse sub-model is parametrized as the Bidirectional-HRED model (Serban et al. 2016) with 1000, 1000, and 2000 hidden units respectively for the coarse-level *encoder*, *context*, and *decoder* RNNs. The natural language sub-model is parametrized as a Bidirectional-HRED model with 500, 1000, and 2000 hidden units respectively for the natural language *encoder*, *context*, and *decoder* RNNs. The *coarse prediction encoder* GRU RNN has 500 hidden units.

Evaluation Methods

It has long been known that accurate evaluation of dialogue system responses is difficult (Schatzmann, Georgila, and Young 2005). Liu et al. (2016) have recently shown that all automatic evaluation metrics adopted for such evaluation, including word overlap-based metrics such as BLEU and METEOR, have either very low or no correlation with human judgment of system performance. We therefore carry out an in-lab human study to evaluate the model responses. We recruit 5 human evaluators. We show each evaluator between 30 and 40 dialogue contexts with the ground truth response, and 4 candidate responses (HRED, HRED + Activity-Entity Features, MrRNN Noun, and MrRNN Activity-Entity). For each example, we ask the evaluators to compare the candidate responses to the ground truth response and dialogue context, and rate them for fluency and relevancy on a scale $0-4$. This setup is very similar to the evaluation setup used by Koehn and Monz (2006), and comparable to Liu et al. (2016). Further details are given in the appendix.

We further propose a new set of evaluation metrics for the Ubuntu domain, in order to ease reproducibility and facilitate future evaluations. These metrics compare the activities and entities in the responses generated by the model with those of the ground truth responses. The assumption is that if a model generates responses with the same meaning as the ground truth responses – including expert responses, which often lead to solving the user's problem – then the model performs well. To compute the metrics, first the ground truth and model responses are mapped to their respective activity-entity representations using the automatic procedure described earlier. Then, the overlap between their activities and entities are measured according to F1-score. Because the activities and entities reflect the principal instructions given in a response – which are key to resolving the technical problem – these metrics should correlate well with solving user problems.

Table 1: Ubuntu evaluation using F1 metrics w.r.t. activities and entities on ground truth utterances (mean scores $\pm$ 90% confidence intervals), and human fluency (F) and human relevancy (R) scores given on a scale 0-4 (* indicates scores significantly different from baseline models at 90% confidence)

	Activity	**Entity**	**Human Eval.**	
Model	**F1**	**F1**	**F**	**R**
LSTM	1.18 ± 0.18	0.87 ± 0.15	-	-
HRED	4.34 ± 0.34	2.22 ± 0.25	2.98	1.01
VHRED	4.63 ± 0.34	2.53 ± 0.26	-	-
HRED + Act.-Ent.	5.46 ± 0.35	2.44 ± 0.26	2.96	0.75
MrRNN Noun	4.04 ± 0.33	$\mathbf{6.31 \pm 0.42}$	$\mathbf{3.48}^*$	$\mathbf{1.32}^*$
MrRNN Act.-Ent.	$\mathbf{11.43 \pm 0.54}$	3.72 ± 0.33	$\mathbf{3.42}^*$	1.04

Results

The results are given in Table 1. The MrRNNs perform substantially better than the baseline models w.r.t. both the human evaluation study and automatic evaluation metrics. MrRNN with noun representations obtains an F1 entity score at 6.31, while the baseline models obtain less than half F1 scores between $0.87 - 2.53$. Further, human evaluators consistently rate its fluency and relevancy significantly higher than all the baseline models. MrRNN with activity representations performs obtains an F1 activity score at 11.43, while all other models obtain less than half F1 activity scores between $1.18 - 5.46$. It also performs substantially better than the baseline models w.r.t. the F1 entity score. In agreement with these results, human evaluators rate its fluency substantially higher than the baseline models. However, its relevancy is rated only slightly higher than HRED. We believe this effect is caused by the human evaluators focusing excessively on software entities in the dialogue responses – rather than focusing on on the actions/activities, which are critical to solving the actual technical problem posed by the user. Furthermore, the MrRNNs outperform the HRED

Table 2: Model response examples. The arrows indicate change of turn.

Dialogue Context	Model Response	Model Coarse Response
Hello I tried to install ubuntu studio but I get W : Failed to fetch <url >404 Not Found when apt-get update → the ppa you added is not online	**MrRNN Act. -Ent.:** try a different mirror **MrRNN Noun:** something **VHRED:** but I dont know how to add it **HRED:** how do I do that ?	**MrRNN Act. -Ent.:** no_tenses try_activity no_cmd **MrRNN Noun:** past_present_tenses ubuntu studio <url>
hi community.. is there any difference between " /.bashrc " " /.bashrc. local " → yes, they're different files :-) the default /.bashrc sources /.bashrc. local → sorry I could undersatnd... If I write on terminal " gedit /.bashrc " it opens a file but how I open " /.bashrc. local " → " gedit /.bashrc. local "... → but it just open blank file nothing inside it	**MrRNN Act.-Ent.:** open up gedit /.bashrc, there is a file called " something " **MrRNN Noun:** the file is empty.. **VHRED:** it's in /.bashrc or /.bashrc **HRED:** what are you trying to do ?	**MrRNN Act.-Ent.:** present_tenses open_activity gedit_entity no_cmd **MrRNN Noun:** present_tenses community difference
Hey guys what do you in general use for irc something ubuntu xchat or xchat-gnome ? → without -gnome. that is just cut down → you mean drop xchat-gnome and go with xchat ?	**MrRNN Act. -Ent.:** im using xchat right now **MrRNN Noun:** what is xchat-gnome ? **VHRED:** correct **HRED:** yes	**MrRNN Act.-Ent.:** present_tenses use_activity xchat_entity no_cmd **MrRNN Noun:** present_tenses guys general irc ubuntu xchat xchatgnome
when setting up rules with iptables command only writes changes this file " etciptables. rules "? i ask this so i can backup before messing anything → sudo iptables-save something . dat to backup your rules restore with sudo iptables-restore < something . dat	**MrRNN Act. -Ent.:** I don't see any reason why you need iptables to do that **MrRNN Noun:** are you using ubuntu ? **VHRED:** thx **HRED:** thanks	**MrRNN Act.-Ent.:** present_tenses look_activity iptables_entity no_cmd **MrRNN Noun:** present_tenses rules iptables command file <path>

baseline augmented activity-entity features across all metrics. This indicates that the hierarchical generation process helps MrRNNs generate responses by adding high-level, compositional structure. Overall, the results indicate that the MrRNNs have learned to model high-level, goal-oriented sequential structure in the Ubuntu domain.

Model responses are shown in Table 2 together with the generated coarse responses. The MrRNN responses are generally very coherent and topic-oriented. In contrast, responses by the other baselines tend to be off-topic or very generic (Li et al. 2016; Serban et al. 2016). This suggests that the hierarchical generation process and factorial representation of the coarse sequences help MrRNNs take dialogue context into account and generalize to new examples. In particular, MrRNN with activity-entity representation appears qualitatively to give more goal-oriented instructions in comparison to MrRNN with noun representation. This is also shown by the goal-oriented coarse responses generated by the coarse sub-model. This observation indicates that the hierarchical generation process helps MrRNN generate responses with more high-level, compositional structure.

Related Work

Li et al. (2016) proposed a model ranking candidate responses according to a mutual information criterion, to incorporate dialogue context efficiently and generate on-topic responses. In comparison, MrRNNs generate on-topic responses without any additional response selection criterion, by jointly modeling high-level and low-level semantics of the dialogue.

More generally, MrRNNs are related to hierarchical RNNs such as the Clockwork RNN (Koutnik et al. 2014) and memory networks (Weston, Chopra, and Bordes 2015; Graves, Wayne, and Danihelka 2014; Kumar et al. 2016). These models also aim to learn high-level representations, but they lack a hierarchical generation process.

Finally, MrRNNs are related to non-neural network latent variable models for dialogue, such as that of Zhai and Williams (2014) and He et al. (2016). Learning task-specific and temporal abstractions for dialogue was also investigated in earlier work by Bangalore et al. (2008), by Crook et al. (2009) and others. MrRNN contributes to this line of work by explicitly modeling the sequence of activities and entities.

Discussion

Current sequence-to-sequence models for dialogue do not effectively take dialogue context into account and cannot generate meaningful and diverse on-topic responses. To address these problems, we have proposed a new class of models called multiresolution recurrent neural networks (MrRNNs) MrRNNs model dialogue through a two-level hierarchical generation process over high-level coarse sequences and natural language utterances. The coarse sequences can incorporate task-specific knowledge and follow a latent stochastic process with a factorial representation, which helps generalization. The hierarchical generation process allows MrRNNs to remember and reason over long-term context, and to generate appropriate responses with compositional structure. Combined with an automatic extraction procedure, the models are trained by optimizing the joint log-likelihood over the sequences at each level. We have applied MrRNNs to dialogue response generation for the Ubuntu technical support domain. We have evaluated them through a human evaluation study and via automatic evaluation metrics. According to both human evaluators and automatic evaluation, MrRNNs improve substantially over competing models. Thus, by representing information at different levels of abstraction and jointly optimizing the generation process across abstraction levels, MrRNNs are able to generate more fluent, relevant and goal-oriented responses. The results indicate that it is not simply a matter of adding additional features for prediction – MrRNNs outperform a competitive model augmented with past coarse sequences as as features – rather, it is the combination of representation and generation at multiple levels that yields the improvements.

MrRNNs are a general framework applicable to any problem where coarse abstractions are available. We conjecture

that MrRNNs may be effective in broader natural language generation tasks, such as generating prose and persuasive argumentation, and other tasks involving discrete sequences, such as music composition.

Acknowledgments The authors thank Ryan Lowe, Michael Noseworthy, Caglar Gulcehre, Sungjin Ahn, Harm de Vries, Song Feng, Sarath Chandar and On Yi Ching for helping with the human study. The authors thank Caglar Gulcehre, Orhan Firat, Ramesh Nallapati and Sara Rosenthal for helpful feedback.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Bangalore, S.; Di Fabbrizio, G.; and Stent, A. 2008. Learning the structure of task-driven human–human dialogs. *IEEE Transactions on Audio, Speech, and Language Processing* 16(7):1249–1259.

Bengio, Y.; Ducharme, R.; Vincent, P.; and Janvin, C. 2003. A neural probabilistic language model. *JMLR* 3:1137–1155.

Bowman, S. R.; Vilnis, L.; Vinyals, O.; Dai, A. M.; Jozefowicz, R.; and Bengio, S. 2016. Generating sentences from a continuous space. *CoNLL*.

Cho, K., et al. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*.

Chung, J.; Kastner, K.; Dinh, L.; Goel, K.; Courville, A.; and Bengio, Y. 2015. A recurrent latent variable model for sequential data. In *NIPS*.

Crook, N.; Granell, R.; and Pulman, S. 2009. Unsupervised classification of dialogue acts using a dirichlet process mixture model. In *SIGDIAL*.

Fabius, O., and van Amersfoort, J. R. 2015. Variational recurrent auto-encoders. *ICLR, Workshop Papers*.

Gorin, A. L.; Riccardi, G.; and Wright, J. H. 1997. How may i help you? *Speech communication* 23(1):113–127.

Graves, A.; Wayne, G.; and Danihelka, I. 2014. Neural turing machines. *arXiv:1410.5401*.

Graves, A. 2012. Sequence transduction with recurrent neural networks. In *ICML, Representation Learning Workshop*.

He, Z.; Liu, X.; Lv, P.; and Wu, J. 2016. Hidden softmax sequence model for dialogue structure analysis. In *ACL*.

Hinton, G., et al. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE* 29(6):82–97.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8).

Ji, Y.; Haffari, G.; and Eisenstein, J. 2016. A latent variable recurrent neural network for discourse relation language models. In *NAACL-HLT*.

Johnson, M. J.; Duvenaud, D.; Wiltschko, A. B.; Datta, S. R.; and Adams, R. P. 2016. Structured VAEs: Composing probabilistic graphical models and variational autoencoders. *arXiv:1603.06277*.

Kannan, A.; Kurach, K.; Ravi, S.; Kaufmann, T.; Tomkins, A.; Miklos, B.; Corrado, G.; Lukács, L.; Ganea, M.; et al. 2016. Smart reply: Automated response suggestion for email. In *ACM SIGKDD*.

Kingma, D., and Ba, J. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Koehn, P., and Monz, C. 2006. Manual and automatic evaluation of machine translation between european languages. In *ACL, Workshop on Statistical Machine Translation*, 102–121.

Koutnik, J.; Greff, K.; Gomez, F.; and Schmidhuber, J. 2014. A Clockwork RNN. In *ICML*.

Kumar, A.; Irsoy, O.; Su, J.; Bradbury, J.; English, R.; Pierce, B.; Ondruska, P.; Gulrajani, I.; and Socher, R. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *ICML*.

Li, J.; Galley, M.; Brockett, C.; Gao, J.; and Dolan, B. 2016. A diversity-promoting objective function for neural conversation models. In *NAACL*.

Liu, C.-W.; Lowe, R.; Serban, I. V.; Noseworthy, M.; Charlin, L.; and Pineau, J. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *EMNLP*.

Lowe, R.; Pow, N.; Serban, I.; and Pineau, J. 2015. The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. In *SIGDIAL*.

Markoff, J., and Mozur, P. 2015. For sympathetic ear, more chinese turn to smartphone program. *NY Times*.

Mikolov, T., et al. 2010. Recurrent neural network based language model. In *INTERSPEECH*.

Pascanu, R.; Mikolov, T.; and Bengio, Y. 2012. On the difficulty of training recurrent neural networks. In *ICML*.

Ritter, A.; Cherry, C.; and Dolan, W. B. 2011. Data-driven response generation in social media. In *EMNLP*.

Schatzmann, J.; Georgila, K.; and Young, S. 2005. Quantitative evaluation of user simulation techniques for spoken dialogue systems. In *SIGDIAL*.

Serban, I. V.; Sordoni, A.; Bengio, Y.; Courville, A. C.; and Pineau, J. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*.

Serban, I. V.; Sordoni, A.; Lowe, R.; Charlin, L.; Pineau, J.; Courville, A.; and Bengio, Y. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *AAAI*.

Shang, L.; Lu, Z.; and Li, H. 2015. Neural responding machine for short-text conversation. In *ACL-IJCNLP*.

Singh, S.; Litman, D.; Kearns, M.; and Walker, M. 2002. Optimizing dialogue management with reinforcement learning: Experiments with the NJFun system. *JAIR* 16:105–133.

Sordoni, A.; Bengio, Y.; Vahabi, H.; Lioma, C.; Simonsen, J. G.; and Nie, J.-Y. 2015a. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *CIKM*.

Sordoni, A.; Galley, M.; Auli, M.; Brockett, C.; Ji, Y.; Mitchell, M.; Nie, J.-Y.; Gao, J.; and Dolan, B. 2015b. A neural network approach to context-sensitive generation of conversational responses. In *NAACL-HLT*.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints* abs/1605.02688.

Weston, J.; Chopra, S.; and Bordes, A. 2015. Memory networks. In *ICLR*.

Young, S. 2000. Probabilistic methods in spoken–dialogue systems. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 358(1769).

Zhai, K., and Williams, J. D. 2014. Discovering latent structure in task-oriented dialogues. In *ACL*.