What to talk about and how? Selective Generation using LSTMs with Coarse-to-Fine Alignment

Hongyuan Mei  
UChicago, TTI-Chicago  
hongyuan@uchicago.edu

Mohit Bansal  
TTI-Chicago  
mbansal@ttic.edu

Matthew R. Walter  
TTI-Chicago  
mwalter@ttic.edu

Abstract
We propose an end-to-end, domain-independent neural encoder-aligner-decoder model for selective generation, i.e., the joint task of content selection and surface realization. Our model first encodes a full set of over-determined database event records via an LSTM-based recurrent neural network, then utilizes a novel coarse-to-fine aligner to identify the small subset of salient records to talk about, and finally employs a decoder to generate free-form descriptions of the aligned, selected records. Our model achieves the best selection and generation results reported to-date (with 59\% relative improvement in generation) on the benchmark WEATHER-GOV dataset, despite using no specialized features or linguistic resources. Using an improved $k$-nearest neighbor beam filter helps further. We also perform a series of ablations and visualizations to elucidate the contributions of our key model components. Lastly, we evaluate the generalizability of our model on the ROBOCUP dataset, and get results that are competitive with or better than the state-of-the-art, despite being severely data-starved.

1 Introduction

We consider the important task of producing a natural language description of a rich world state represented as an over-determined database of event records. This task, which we refer to as selective generation, is often formulated as two subproblems: content selection, which involves choosing a subset of relevant records to talk about from the exhaustive database, and surface realization, which is concerned with generating natural language descriptions for this subset. Learning to perform these tasks jointly is challenging due to the uncertainty in deciding which records are relevant, the complex dependencies between selected records, and the multiple ways in which these records can be described.

Previous work has made significant progress on this task (Chen and Mooney, 2008; Angeli et al., 2010; Kim and Mooney, 2010; Konstas and Lapata, 2012). However, most approaches solve the two content selection and surface realization sub-tasks separately, use manual domain-dependent resources (e.g., semantic parsers) and features, or employ template-based generation. This limits domain adaptability and reduces coherence. We take an alternative, neural encoder-aligner-decoder approach to free-form selective generation that jointly performs content selection and surface realization, without using any specialized features, resources, or generation templates. This enables our approach to generalize to new domains. Further, our memory-based model captures the long-range contextual dependencies among records and descriptions, which are integral to this task (Angeli et al., 2010).

We formulate our model as an encoder-aligner-decoder framework that uses recurrent neural networks with long short-term memory units (LSTM-RNNs) (Hochreiter and Schmidhuber, 1997) together with a coarse-to-fine aligner to select and “translate” the rich world state into a natural language description. Our model first encodes the full set of over-determined event records using a bidirectional LSTM-RNN. A novel coarse-to-fine aligner then reasons over multiple abstractions of the input to decide which of the records to discuss. The model next employs an LSTM decoder to gen-

\footnote{By “over-determined”, we mean that there are extraneous and redundant records present in the database.}
erate natural language descriptions of the selected records.

The use of LSTMs, which have proven effective for similar long-range generation tasks (Sutskever et al., 2014; Vinyals et al., 2015b; Karpathy and Fei-Fei, 2015), allows our model to capture the long-range contextual dependencies that exist in selective generation. Further, the introduction of our proposed variation on alignment-based LSTMs (Bahdanau et al., 2014; Xu et al., 2015) enables our model to learn to perform content selection and surface realization jointly, by aligning each generated word to an event record during decoding. Our novel coarse-to-fine aligner avoids searching over the full set of over-determined records by employing two stages of increasing complexity: a pre-selector and a refiner acting on multiple abstractions (low- and high-level) of the record input. The end-to-end nature of our framework has the advantage that it can be trained directly on corpora of record sets paired with natural language descriptions, without the need for ground-truth content selection.

We evaluate our model on a benchmark weather forecasting dataset (WEATHERGOV) and achieve the best results reported to-date on content selection (12% relative improvement in F-1) and language generation (59% relative improvement in BLEU), despite using no domain-specific resources. We also perform a series of ablations and visualizations to elucidate the contributions of the primary model components, and also show improvements with a simple, k-nearest neighbor beam filter approach. Finally, we demonstrate the generalizability of our model by directly applying it to a benchmark sportscasting dataset (ROBOCUP), where we get results competitive with or better than state-of-the-art, despite being extremely data-starved.

2 Related Work

Selective generation is a task where a natural language description is produced for a salient subset of a rich world state represented as an over-determined database of event records. A good deal of attention in this area has been paid to the individual content selection and selective realization subproblems. With regards to the former, Barzilay and Lee (2004) model the content structure from unannotated documents and apply it to the application of text summarization. Barzilay and Lapata (2005) treat content selection as a collective classification problem and simultaneously optimize the local label assignment and their pairwise relations. Liang et al. (2009) address the related task of aligning a set of records to given textual description clauses. They propose a generative semi-Markov alignment model that jointly segments text sequences into utterances and associates each to the corresponding record.

Surface realization is often treated as a problem of producing text according to a given representation (Reiter et al., 2000). Walker et al. (2001) and Stent et al. (2004) design trainable sentence planners to generate sentences (and their combinations) for context planning and dialogue, relying upon various linguistics features. Soricut and Marcu (2006) propose a language generation system that uses the WIDL-representation, a formalism used to compactly represent probability distributions over finite sets of strings. Wong and Mooney (2007) and Lu and Ng (2011) use synchronous context-free grammars to generate natural language sentences from formal meaning representations. Similarly, Belz (2008) employs probabilistic context-free grammars to perform surface realization. Other effective approaches include the use of tree conditional random fields (Lu et al., 2009) and template extraction within a log-linear framework (Angeli et al., 2010).

Recent work seeks to solve the full selective generation problem through a single framework. Chen and Mooney (2008) and Chen et al. (2010) learn alignments between comments and their corresponding event records using a translation model for parsing and generation. Kim and Mooney (2010) implement a two-stage framework that decides what to discuss using a combination of the methods of Lu et al. (2008) and Liang et al. (2009), and then produces the text based on the generation system of Wong and Mooney (2007).

Angeli et al. (2010) propose a unified concept-to-text model that treats joint content selection and surface realization as a sequence of local decisions represented by a log-linear model. Similar to other work, they train their model using external alignments from Liang et al. (2009). Generation then follows as inference over this model, where they first choose an event record, then the record’s fields (i.e.,
attributes), and finally a set of templates that they then fill in with words for the selected fields. Their ability to model long-range dependencies relies on their choice of features for the log-linear model, while the template-based generation further employs some domain-specific features for fluent output.

Konstas and Lapata (2012) propose an alternative method that simultaneously optimizes the content selection and surface realization problems. They employ a probabilistic context-free grammar that specifies the structure of the event records, and then treat generation as finding the best derivation tree according to this grammar. However, their method still selects and orders records in a local fashion via a Markovized chaining of records. Konstas and Lapata (2013) improve upon this approach with global document representations. However, this approach also requires alignment during training, which they estimate using the method of Liang et al. (2009).

We treat the problem of selective generation as end-to-end learning via a recurrent neural network encoder-aligner-decoder model, which enables us to jointly learn content selection and surface realization directly from database-text pairs, without the need for an external aligner or ground-truth selection labels. The use of LSTM-RNNs enables our model to capture the long-range dependencies that exist among the records and natural language output. Additionally, the model does not rely on any manually-selected or domain-dependent features, templates, or parsers, and is thereby generalizable. The alignment-RNN approach has recently proven successful for generation-style tasks, e.g., machine translation (Bahdanau et al., 2014) and image captioning (Xu et al., 2015). Since selective generation requires identifying the small number of salient records among an over-determined database, we avoid performing exhaustive search over the full record set, and instead propose a novel coarse-to-fine aligner that divides the search complexity into pre-selection and refinement stages.

3 Task Definition

We consider the problem of generating a natural language description for a rich world state specified in terms of an over-determined set of records (database). This problem requires deciding which of the records to discuss (content selection) and how to discuss them (surface realization). Training data consists of scenario pairs \( (r^{(i)}, x^{(i)}) \) for \( i = 1, 2, \ldots, n \), where \( r^{(i)} \) is the complete set of records and \( x^{(i)} \) is the natural language description (Fig. 1). At test time, only the records are given. We evaluate our model in the context of two publicly-available benchmark selective generation datasets.

**WeatherGov** The weather forecasting dataset (see Fig. 1(a)) of Liang et al. (2009) consists of 29528 scenarios, each with 36 weather records (e.g., temperature, sky cover, etc.) paired with a natural language forecast (28.7 avg. word length).

**RoboCup** We evaluate our model’s generalizability on the sportscasting dataset of Chen and Mooney (2008), which consists of only 1539 pairs of temporally ordered robot soccer events (e.g., pass, score) and commentary drawn from the four-game 2001–2004 RoboCup finals (see Fig. 1(b)). Each scenario contains an average of 2.4 event records and a 5.7 word natural language commentary.

4 The Model

We formulate selective generation as inference over a probabilistic model \( P(x_{1:T} | r_{1:N}) \), where
\[ \arg \max_{x_{1:T}} P(x_{1:T} | r_{1:N}) \]

\[ = \arg \max_{x_{1:T}} \prod_{t=1}^{T} P(x_t | x_{0:t-1}, r_{1:N}) \quad \text{(1b)} \]

The goal of inference is to generate a natural language description for a given set of records. An effective means of learning to perform this generation is to use an encoder-aligner-decoder architecture with a recurrent neural network, which has proven effective for related problems in machine translation (Bahdanau et al., 2014) and image captioning (Xu et al., 2015). We propose a variation on this general model with novel components that are well-suited to the selective generation problem.

Our model (Fig. 2) first encodes each input record \( r_j \) into a hidden state \( h_j \) with \( j \in \{1, \ldots, N\} \) using a bidirectional recurrent neural network (RNN). Our novel coarse-to-fine aligner then acts on a concatenation \( m_j \) of each record and its hidden state as multi-level representation of the input to compute the selection decision \( z_t \) at each decoding step \( t \). The model then employs an RNN decoder to arrive at the word likelihood \( P(x_t | x_{0:t-1}, r_{1:N}) \) as a function of the multi-level input and the hidden state of the decoder \( s_{t-1} \) at time step \( t-1 \). In order to model the long-range dependencies among the records and descriptions (which is integral to effectively performing selective generation (Angeli et al., 2010; Konstas and Lapata, 2012; Konstas and Lapata, 2013)), our model employs LSTM units as the nonlinear encoder and decoder functions.

**Encoder** Our LSTM-RNN encoder (Fig. 2) takes as input the set of event records represented as a sequence \( r_{1:N} = (r_1, r_2, \ldots, r_N) \) and returns a sequence of hidden annotations \( h_{1:N} = (h_1, h_2, \ldots, h_N) \), where the annotation \( h_j \) summarizes the record \( r_j \). This results in a representation that models the dependencies that exist among the records in the database. We adopt an encoder architecture similar to that of Graves et al. (2013)

\[
\begin{pmatrix}
  i_j^e \\
  f_j^e \\
  o_j^e \\
  g_j^e
\end{pmatrix} =
\begin{pmatrix}
  \sigma \\
  \sigma \\
  \sigma \\
  \tanh
\end{pmatrix}
T^e
\begin{pmatrix}
  r_j \\
  h_{j-1}
\end{pmatrix}
\]

\[ c_j^e = f_j^e \odot c_{j-1}^e + i_j^e \odot g_j^e \quad \text{(2b)} \]

\[ h_j = o_j^e \odot \tanh(c_j^e) \quad \text{(2c)} \]

where \( T^e \) is an affine transformation, \( \sigma \) is the logistic sigmoid that restricts its input to \([0, 1]\), \( i_j^e \), \( f_j^e \), and \( o_j^e \) are the input, forget, and output gates of the LSTM, respectively, and \( c_j^e \) is the memory cell activation vector. The memory cell \( c_j^e \) summarizes the LSTM’s previous memory \( c_{j-1}^e \) and the current input, which are modulated by the forget and input gates, respectively. Our encoder operates bidirectionally, encoding the records in both the forward and backward directions, which provides a better summary of the input records. In this way, the hidden annotations \( h_j = (h_j^f, h_j^b) \) concatenate forward \( h_j \) and backward \( h_j \) annotations, each determined using Equation (2c).

**Coarse-to-Fine Aligner** Having encoded the input records \( r_{1:N} \) to arrive at the hidden annotations \( h_{1:N} \), the model then seeks to select the content at each time step \( t \) that will be used for generation. Our
model performs content selection using an extension of the alignment mechanism proposed by Bahdanau et al. (2014), which allows for selection and generation that is independent of the ordering of the input.

In selective generation, the given set of event records is over-determined with only a small subset of salient records being relevant to the output natural language description. Standard alignment mechanisms limit the accuracy of selection and generation by scanning the entire range of over-determined records. In order to better address the selective generation task, we propose a coarse-to-fine aligner that prevents the model from being distracted by non-salient records. In order to better address the selective generation task, we propose a coarse-to-fine aligner that prevents the model from being distracted by non-salient records. In order to better address the selective generation task, we propose a coarse-to-fine aligner that prevents the model from being distracted by non-salient records. Our coarse-to-fine aligner avoids searching over the full set of over-determined records by using two stages of increasing complexity: a pre-selector and refiner (Fig. 2). The pre-selector first assigns to each record a probability $p_j$ of being selected, while the standard aligner computes the alignment likelihood $w_{tj}$ over all the records at each time step $t$ during decoding. Next, the refiner produces the final selection decision by re-weighting the aligner weights $w_{tj}$ with the pre-selector probabilities $p_j$:

$$p_j = \text{sigmoid} \left( q^\top \tanh(Pm_j) \right)$$

$$\beta_{tj} = v^\top \tanh(Ws_{t-1} + Um_j)$$

$$w_{tj} = \exp(\beta_{tj}) / \sum_j \exp(\beta_{tj})$$

$$\alpha_{tj} = p_j w_{tj} / \sum_j p_j w_{tj}$$

$$z_t = \sum_j \alpha_{tj} m_j$$

where $P$, $q$, $U$, $W$, $v$ are learned parameters. Ideally, the selection decision would be based on the highest-value alignment $z_t = m_k$ where $k = \arg \max_j \alpha_{tj}$. However, we use the weighted average (Eqn. 3e) as its soft approximation to maintain differentiability of the entire architecture.

The pre-selector assigns large values ($p_j > 0.5$) to a small subset of salient records and small values ($p_j < 0.5$) to the rest. This modulates the standard aligner, which then has to assign a large weight $w_{tj}$ in order to select the $j$-th record at time $t$. In this way, the learned prior $p_j$ makes it difficult for the alignment (attention) to be distracted by non-salient records. Further, we can relate the output of the pre-selector to the number of records that are selected. Specifically, the output $p_j$ expresses the extent to which the $j$-th record should be selected. The summation $\sum_{j=1}^N p_j$ can then be regarded as a real-valued approximation to the total number of pre-selected records (denoted as $\gamma$), which we regularize towards, based on validation (see Eqn. 5).

Decoder Our architecture uses an LSTM decoder that takes as input the current context vector $z_t$, the last word $x_{t-1}$, and the LSTM’s previous hidden state $s_{t-1}$. The decoder outputs the conditional probability distribution $P_{x,t} = P(x_t|x_{0:t-1}, r_{1:N})$ over the next word, represented as a deep output layer (Pascu et al., 2014),

$$P_{x,t} = \text{softmax} \left( \begin{pmatrix} i_t^d \\ f_t^d \\ c_t^d \\ g_t^d \\ o_t^d \end{pmatrix} \right)$$

where $E$ (an embedding matrix), $L_0$, $L_s$, and $L_z$ are parameters to be learned.

Training and Inference We train the model using the database-record pairs $(r_{1:N}, x_{1:T})$ from the training corpora so as to maximize the likelihood of the ground-truth language description $x_{1:T}^*$ (Eqn. 1). Additionally, we introduce a regularization term $\left( \sum_{j=1}^N p_j - \gamma \right)^2$ that enables the model to influence the pre-selector weights based on the aforementioned relationship between the output of the pre-selector and the number of selected records. Moreover, we also introduce the term $(1.0 - \max_j(p_j))$, which accounts for the fact that at least one record should be pre-selected. Note that when $\gamma$ is equal to $N$, the pre-selector is forced to select all the records.
\[ (p_j = 1.0 \text{ for all } j), \text{ and the coarse-to-fine alignment reverts to the standard alignment introduced by Bahdanau et al. (2014). Together with the negative log-likelihood of the ground-truth description } x_{1:T}^*, \text{ our loss function becomes} \]

\[ L = -\log P(x_{1:T}^*|r_{1:N}) + G \]  
\[ = -\sum_{t=1}^{T} \log P(x_t^*|x_{0:t-1}, r_{1:N}) + G \]

\[ G = \left( \sum_{j=1}^{N} p_j - \gamma \right)^2 + \left( 1 - \max(p_j) \right) \]

Having trained the model, we generate the natural language description by finding the maximum a posteriori words under the learned model (Eqn. 1).\footnote{Numerical values are also generated exactly as any other token in the vocabulary.} For inference, we perform greedy search starting with the first word \( x_1 \). Beam search offers a way to perform approximate joint inference — however, we empirically found that beam search does not perform any better than greedy search on the datasets that we consider, an observation that is shared with previous work (Angeli et al., 2010). We later discuss an alternative \( k \)-nearest neighbor-based beam filter (see Sec 6.2).

5 Experimental Setup

Datasets We analyze our model on the benchmark \textsc{WeatherGov} dataset, and use the data-starved \textsc{RoboCup} dataset to demonstrate the model’s generalizability. Following Angeli et al. (2010), we use \textsc{WeatherGov} training, development, and test splits of size 25000, 1000, and 3528, respectively. For \textsc{RoboCup}, we follow the evaluation methodology of previous work (Chen and Mooney, 2008), performing three-fold cross-validation whereby we train on three games (approximately 1000 scenarios) and test on the fourth. Within each split, we hold out 10\% of the training data as the development set to tune the early-stopping criterion and \( \gamma \). We then report the standard average performance (weighted by the number of scenarios) over these four splits.

Dataset Processing In this section, we present the implementation details regarding our data preprocessing. We use \textsc{WeatherGov} as an example here, since it is our primary dataset, and the same recipe is followed for \textsc{RoboCup}.

For tokenization of the textual descriptions, we simply treat as token each string unit delimited by a space, which includes regular words (“sunny”), punctuation (“,”), and numerical values (“20”). A special token is added to represent the beginning and end of the entire textual description. This operation results in a vocabulary of size 338, and we did not filter out any rare tokens. Moreover, in this setup, numerical values are also generated as any other token during decoding period.

For event record representation, we represent each event as a fixed-length vector, concatenated by multiple “attribute (field) vectors”. Each attribute vector represents either a 1) record type (e.g., “rain-Chance”) with a one-hot vector, 2) record time slot (e.g., “06:00–21:00”) with a one-hot vector, 3) record mode (e.g., “SSE”) with a one-hot vector, or, 4) record value (e.g., “20”) with a 0-1 vector. The 0-1 vector for record value is simply the signed binary representation of this number. We choose the usage of 0-1 binary representation vectors for numbers because it allows us to share binning-style information between nearby numbers (whereas a one-hot vector is sparse).

Training Details On \textsc{WeatherGov}, we lightly tune the number of hidden units and \( \gamma \) on the development set according to the generation metric (BLEU), and choose 500 units from \{250, 500, 750\} and \( \gamma = 8.5 \) from \{6.5, 7.5, 8.5, 10.5, 12.5\}. For \textsc{RoboCup}, we only tune \( \gamma \) on the development set and choose \( \gamma = 5.0 \) from the set \{1.0, 2.0, \ldots, 6.0\}. However, we do not retrain the number of hidden units on \textsc{RoboCup}. For each iteration, we randomly sample a mini-batch of 100 scenarios during back-propagation and use Adam (Kingma and Ba, 2015) for optimization. Training typically converges within 30 epochs. We select the model according to the BLEU score on the development set.\footnote{We implement our model in Theano (Bergstra et al., 2010; Bastien et al., 2012) and will make the code publicly available.}

Evaluation Metrics We consider two metrics as a means of evaluating the effectiveness of our model on the two selective generation subproblems. For content selection, we use the F-1 score of the set of
selected records as defined by the harmonic mean of precision and recall with respect to the ground-truth selection record set. We define the set of selected records as consisting of the record with the largest selection weight \( \alpha_{kl} \) computed by our aligner at each decoding step \( t \).

We evaluate the quality of surface realization using the BLEU score\(^6\) (a 4-gram matching-based precision) (Papineni et al., 2001) of the generated description with respect to the human-created reference. To be comparable to previous results on WEATHERGOV, we also consider a modified BLEU score (cBLEU) that does not penalize numerical deviations of at most five (Angeli et al., 2010) (i.e., to not penalize “low around 58” compared to a reference “low around 60”). On ROBOCUP, we also evaluate the BLEU score in the case that ground-truth content selection is known (sBLEU\(_G\)), to be comparable to previous work.

6 Results and Analysis

We analyze the effectiveness of our model on the benchmark WEATHERGOV (as primary) and ROBOCUP (as generalization) datasets. We also present several ablations to illustrate the contributions of the primary model components.

6.1 Primary Results (WEATHERGOV)

We report the performance of content selection and surface realization using F-1 and two BLEU scores (standard sBLEU and the customized cBLEU of Angeli et al. (2010)), respectively (Sec. 5). Table 1 compares our test results against previous methods that include KL12 (Konstas and Lapata, 2012), KL13 (Konstas and Lapata, 2013), and ALK10 (Angeli et al., 2010). Our method achieves the best results reported to-date on all three metrics, with relative improvements of 11.94% (F-1), 58.88% (sBLEU), and 36.68% (cBLEU) over the previous state-of-the-art.

6.2 Beam Filter with \( k \)-Nearest Neighbors

We perform greedy search as an approximation to full inference over the set of decision variables (Eqn. 1). We considered beam search as an alternative, but as with previous work on this dataset (Angeli et al., 2010), we found that greedy search still yields better BLEU performance (Table 2).

<table>
<thead>
<tr>
<th>Beam width ( M )</th>
<th>1</th>
<th>2</th>
<th>5</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>dev sBLEU</td>
<td>65.58</td>
<td>64.70</td>
<td>57.02</td>
<td>47.07</td>
</tr>
<tr>
<td>dev cBLEU</td>
<td>75.78</td>
<td>74.91</td>
<td>65.83</td>
<td>54.19</td>
</tr>
</tbody>
</table>

As an alternative, we consider a beam filter based on a \( k \)-nearest neighborhood. First, we generate the \( M \)-best description candidates (i.e., a beam width of \( M \)) for a given input record set (database) using standard beam search. Next, we find the \( K \) nearest neighbor database-description pairs from the training data, based on the cosine similarity of each neighbor database with the given input record. We then compute the BLEU score for each of the \( M \) description candidates relative to the \( K \) nearest neighbor descriptions (as references) and select the candidate with the highest BLEU score. We tune \( K \) and \( M \) on the development set and report the results in Table 3. Table 4 presents the test results with this tuned setting (\( M = 2, K = 1 \)), where we achieve BLEU scores better than our primary greedy results.

<table>
<thead>
<tr>
<th>( M )</th>
<th>2</th>
<th>5</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>sBLEU</td>
<td>65.99</td>
<td>65.88</td>
<td>65.65</td>
</tr>
<tr>
<td>cBLEU</td>
<td>76.21</td>
<td>76.13</td>
<td>75.98</td>
</tr>
</tbody>
</table>

---

\(^6\) We compute BLEU using the publicly available evaluation provided by Angeli et al. (2010).
id-0: temperature(time=06-21, min=52, mean=63, max=71); id-2: windSpeed(time=06-21, min=8, mean=17, max=23);
id-3: windDir(time=06-21, mode=SSE); id-4: gust(time=06-21, min=0, mean=10, max=30);
id-5: skyCover(time=6-21, mode=50-75); id-10: precipChance(time=06-21, min=19, mean=32, max=73);
id-15: thunderChance(time=13-21, mode=SChc)

Figure 3: An example generation for a set of records from WEATHERGov.

Table 4: k-NN beam filter (test set)

<table>
<thead>
<tr>
<th></th>
<th>Primary</th>
<th>k-NN (M = 2, K = 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sBLEU</td>
<td>cBLEU</td>
</tr>
<tr>
<td></td>
<td>61.01</td>
<td>61.76</td>
</tr>
<tr>
<td></td>
<td>70.39</td>
<td>71.23</td>
</tr>
</tbody>
</table>

6.3 Ablation Analysis (WEATHERGov)

Next, we present several ablations to analyze the contribution of our model components. 

Aligner Ablation First, we evaluate the contribution of our proposed coarse-to-fine aligner by comparing our model with the basic encoder-aligner-decoder model introduced by Bahdanau et al. (2014) (which we originally started with). Table 5 reports the results demonstrating that our two-level aligner yields superior F-1 and BLEU scores relative to a standard aligner. 

Table 5: Coarse-to-fine aligner ablation (dev set)

<table>
<thead>
<tr>
<th></th>
<th>F-1</th>
<th>sBLEU</th>
<th>cBLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>60.35</td>
<td>63.54</td>
<td>74.90</td>
</tr>
<tr>
<td>Coarse-to-fine</td>
<td>76.28</td>
<td>65.58</td>
<td>75.78</td>
</tr>
</tbody>
</table>

Encoder Ablation Next, we consider the effectiveness of the encoder. Table 6 compares the results with and without the encoder on the development set, and demonstrates that there is a significant gain from encoding the event records using the LSTM-RNN. We attribute this improvement to the LSTM-RNN’s ability to capture the relationships that exist among the records, which is known to be essential to selective generation (Barzilay and Lapata, 2005; Angeli et al., 2010).

Table 6: Encoder ablation (dev set)

<table>
<thead>
<tr>
<th></th>
<th>F-1</th>
<th>sBLEU</th>
<th>cBLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>With</td>
<td>76.28</td>
<td>65.58</td>
<td>75.78</td>
</tr>
<tr>
<td>Without</td>
<td>57.45</td>
<td>56.47</td>
<td>68.63</td>
</tr>
</tbody>
</table>

6.4 Qualitative Analysis (WEATHERGov)

Output Examples Fig. 3 shows an example record set with its output description and record-word alignment heat map. As shown, our model learns to align records with their corresponding words (e.g., windDir and “southeast,” temperature and “71,” windSpeed and “wind 10,” and gust and “winds could gust as high as 30 mph”). It also learns the subset of salient records to talk about (matching the ground-truth description perfectly for this example, i.e., a standard BLEU of 100.00). We also see some word-level mismatch, e.g., “cloudy” mis-
aligns to id-0 temp and id-10 precipChance, which we attribute to the high correlation between these types of records ("garbage collection" in Liang et al. (2009)).

**Word Embeddings (Trained & Pretrained)**
Training our decoder has the effect of learning embeddings for the words in the training set (via the embedding matrix $E$ in Eqn. 4). Here, we explore the extent to which these learned embeddings capture semantic relationships among the training words. Table 7 presents nearest neighbor words for some of the common words from the WEATHER-GOV dataset (according to cosine similarity in the embedding space).

<table>
<thead>
<tr>
<th>Word</th>
<th>Nearest neighbor</th>
</tr>
</thead>
<tbody>
<tr>
<td>gusts</td>
<td>gust</td>
</tr>
<tr>
<td>clear</td>
<td>sunny</td>
</tr>
<tr>
<td>isolated</td>
<td>scattered</td>
</tr>
<tr>
<td>southeast</td>
<td>northeast</td>
</tr>
<tr>
<td>storms</td>
<td>winds</td>
</tr>
<tr>
<td>decreasing</td>
<td>falling</td>
</tr>
</tbody>
</table>

We also consider different ways of using pre-trained word embeddings (Mikolov et al., 2013) to bootstrap the quality of our learned embeddings. One approach initializes our embedding matrix with the pre-trained vectors and then refines the embedding based on our training corpus. The second concatenates our learned embedding matrix with the pre-trained vectors in an effort to simultaneously exploit general similarities as well as those learned for the domain. As shown previously for other tasks (Vinyals et al., 2014; Vinyals et al., 2015b), we find that the use of pre-trained embeddings results in negligible improvements (on the development set).

### 6.5 Out-of-Domain Results (ROBOCUP)
We use the ROBOCUP dataset to evaluate the domain-independence of our model. The dataset is severely data-starved with only 1000 (approx.) training pairs, which is much smaller than is typically necessary to train RNNs. This results in higher variance in the trained model distributions, and we thus adopt the standard denoising method of ensembles (Sutskever et al., 2014; Vinyals et al., 2015b; Zaremba et al., 2014).

Table 8: ROBOCUP results

<table>
<thead>
<tr>
<th>Method</th>
<th>F-1</th>
<th>sBLEU</th>
<th>sBLEUG</th>
</tr>
</thead>
<tbody>
<tr>
<td>CM08</td>
<td>72.00</td>
<td>–</td>
<td>28.70</td>
</tr>
<tr>
<td>LJK09</td>
<td>75.70</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>CKM10</td>
<td>79.30</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>ALK10</td>
<td>79.90</td>
<td>–</td>
<td>28.80</td>
</tr>
<tr>
<td>KL12</td>
<td>–</td>
<td>24.88</td>
<td>30.90</td>
</tr>
<tr>
<td>Our model</td>
<td>81.58</td>
<td>25.28</td>
<td>29.40</td>
</tr>
</tbody>
</table>

Following previous work, we perform two experiments on the ROBOCUP dataset (Table 8), the first considering full selective generation and the second assuming ground-truth content selection at test time. On the former, we obtain a standard BLEU score (sBLEU) of 25.28, which exceeds the best score of 24.88 (Konstas and Lapata, 2012). Additionally, we achieve an selection F-1 score of 81.58, which is also the best result reported to-date. In the case of assumed (known) ground-truth content selection, our model attains an sBLEUG score of 29.40, which is competitive with the state-of-the-art.

### 7 Conclusion
We presented an encoder-aligner-decoder model for selective generation that does not use any specialized features, linguistic resources, or generation templates. Our model employs a bidirectional LSTM-RNN model with a novel coarse-to-fine aligner that jointly learns content selection and surface realization. We evaluate our model on the benchmark WEATHERGOV dataset and achieve state-of-the-art selection and generation results. We achieve further improvements via a $k$-nearest neighbor beam filter. We also present several model ablations and visualizations to elucidate the effects of the primary components of our model. Moreover, our model generalizes to a different, data-starved domain (ROBOCUP), where it achieves results competitive with or better than the state-of-the-art.

### Acknowledgments
We thank Gabor Angeli, David Chen, Ioannis Konstas, and the reviewers for their helpful comments.

---

9 We use an ensemble of five randomly initialized models.

10 The Chen and Mooney (2008) sBLEUG result is from Angeli et al. (2010).
References


