End-to-End Content and Plan Selection for Data-to-Text Generation

Sebastian Gehrmann
Harvard SEAS
gehrmann@seas.harvard.edu

Falcon Z. Dai
TTI-Chicago
dai@ttic.edu

Henry Elder
ADAPT
henry.elder@adaptcentre.ie

Alexander M. Rush
Harvard SEAS
srush@seas.harvard.edu

Abstract

Learning to generate fluent natural language from structured data with neural networks has become an common approach for NLG. This problem can be challenging when the form of the structured data varies between examples. This paper presents a survey of several extensions to sequence-to-sequence models to account for the latent content selection process, particularly variants of copy attention and coverage decoding. We further propose a training method based on diverse ensembling to encourage models to learn distinct sentence templates during training. An empirical evaluation of these techniques shows an increase in the quality of generated text across five automated metrics, as well as human evaluation.

1 Introduction

Recent developments in end-to-end learning with neural networks have enabled methods to generate textual output from complex structured inputs such as images and tables. These methods may also enable the creation of text-generation models that are conditioned on multiple key-value attribute pairs. The conditional generation of fluent text poses multiple challenges since a model has to select content appropriate for an utterance, develop a sentence layout that fits all selected information, and finally generate fluent language that incorporates the content. End-to-end methods have already been applied to increasingly complex data to simultaneously learn sentence planning and surface realization but were often restricted by the limited data availability (Wen et al., 2015; Mei et al., 2015; Dušek and Jurčíček, 2016; Lampouras and Vlachos, 2016). The recent creation of datasets such as the E2E NLG dataset (Novikova et al., 2017) provides an opportunity to further advance methods for text generation. In this work, we focus on the generation of language from meaning representations (MR), as shown in Figure 1. This task requires learning a semantic alignment from MR to utterance, wherein the MR can comprise a variable number of attributes.

MR	name: The Golden Palace, eatType: coffee shop, food: Fast food, priceRange: cheap, customer rating: 5 out of 5, area: riverside
Reference	A coffee shop located on the riverside called The Golden Palace, has a 5 out of 5 customer rating. Its price range are fairly cheap for its excellent Fast food.

Figure 1: An example of a meaning representation and utterance pair from the E2E NLG dataset. Each example comprises a set of key-value pairs and a natural language description.

Recently, end-to-end generation has been handled primarily by Sequence-to-sequence (S2S) models (Sutskever et al., 2014; Bahdanau et al., 2014) that encode some information and decode it into a desired format. Extensions for summarization and other tasks have developed a mechanism to copy words from the input into a generated text (Vinyals et al., 2015; See et al., 2017).

We begin with a strong S2S model with copy-mechanism for the E2E NLG task and include methods that can help to control the length of a generated text and how many inputs a model uses (Tu et al., 2016; Wu et al., 2016). Finally,

we also present results of the Transformer architecture (Vaswani et al., 2017) as an alternative S2S variant. We show that these extensions lead to improved text generation and content selection.

We further propose a training approach based on the diverse ensembling technique (Guzman-Rivera et al., 2012). In this technique, multiple models are trained to partition the training data during the process of training the model itself, thus leading to models that follow distinct sentence templates. We show that this approach improves the quality of generated text, but also the robustness of the training process to outliers in the training data.

Experiments are run on the E2E NLG challenge[1]. We show that the application of this technique increases the quality of generated text across five different automated metrics (BLEU, NIST, METEOR, ROUGE, and CIDEr) over the multiple strong S2S baseline models (Dušek and Jurčíček, 2016; Vaswani et al., 2017; Su et al., 2018; Freitag and Roy, 2018). Among 60 submissions to the challenge, our approach ranked first in METEOR, ROUGE, and CIDEr scores, third in BLEU, and sixth in NIST.

2 Related Work

Traditional approaches to natural language generation separate the generation of a sentence plan from the surface realization. First, an input is mapped into a format that represents the layout of the output sentence, for example, an adequate pre-defined template. Then, the surface realization transforms the intermediary structure into text (Stent et al., 2004). These representations often model the hierarchical structure of discourse relations (Walker et al., 2007). Early data-driven approach used phrase-based language models for generation (Oh and Rudnicky, 2000; Mairesse and Young, 2014), or aimed to predict the best fitting cluster of semantically similar templates (Kondadadi et al., 2013). More recent work combines both steps by learning plan and realization jointly using end-to-end trained models (e.g. Wen et al., 2015). Several approaches have looked at generation from abstract meaning representations (AMR), and Peng et al. (2017) apply S2S models to the problem. However, Ferreira et al. (2017) show that S2S models are outperformed by phrase-based machine translation models in small datasets. To address this issue, Konstas et al. (2017) propose a semi-supervised training method that can utilize English sentences outside of the training set to train parts of the model. We address the issue by using copy-attention to enable the model to copy words from the source, which helps to generate out of vocabulary and rare words. We note that end-to-end trained models, including our approach, often do not explicitly model the sentence planning stage, and are thus not directly comparable to previous work on sentence planning. This is especially limiting for generation of complex argument structures that rely on hierarchical structure.

For the task of text generation from simple key-value pairs, as in the E2E task, Juraska et al. (2018) describe a heuristic based on word-overlap that provides unsupervised slot alignment between meaning representations and open slots in sentence plans. This method allows a model to operate with a smaller vocabulary and to be agnostic to actual values in the meaning representations. To account for syntactic structure in templates, Su et al. (2018) describe a hierarchical decoding strategy that generates different part of speech at different steps, filling in slots between previously generated tokens. In contrast, our model uses copy-attention to fill in latent slots inside of learned templates. Juraska et al. (2018) also describe a data selection process in which they use heuristics to filter a dataset to the most natural sounding examples according to a set of rules. Our work aims at the unsupervised segmentation of data such that one model learns the most natural sounding sentence plans.

3 Background: Sequence-to-Sequence Generation

We start by introducing the standard a text-to-text problem and discuss how to map structured data into a sequential form. Let $(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}), \ldots (\mathbf{x}^{(N)}, \mathbf{y}^{(N)}) \in (\mathcal{X}, \mathcal{Y})$ be a set of N aligned source and target sequence pairs, with $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ denoting the ith element in $(\mathcal{X}, \mathcal{Y})$ pairs. Further, let $\mathbf{x} = x_1, \ldots, x_m$ be the sequence of m tokens in the source, and $\mathbf{y} = y_1, \ldots, y_n$ the target sequence of length n. Let $\mathcal{V}$ be the vocabulary of possible tokens, and $[n]$ the list of integers up to n, $[1, \ldots, n]$.

S2S aims to learn a distribution parametrized

[1] http://www.macs.hw.ac.uk/InteractionLab/E2E/

by θ to maximize the conditional probability of $p_\theta(\mathbf{y}|\mathbf{x})$. We assume that the target is generated from left to right, such that $p_\theta(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{n} p_\theta(y_t|\mathbf{y}_{[t-1]}, \mathbf{x})$, and that $p_\theta(y_t|\mathbf{y}_{[t-1]}, \mathbf{x})$ takes the form of an encoder-decoder architecture with attention. The training aims to maximize the log-likelihood of the observed training data.

We evaluate the performance of both the LSTM (Hochreiter and Schmidhuber, 1997) and Transformer (Vaswani et al., 2017) architecture. We additionally experiment with two attention formulations. The first uses a dot-product between the hidden states of the encoder and decoder (Luong et al., 2015). The second uses a multi-layer perceptron with the hidden states as inputs (Bahdanau et al., 2014). We refer to them as *dot* and *MLP* respectively. Since *dot* attention does not require additional parameters, we hypothesize that it performs well in a limited data environment.

In order to apply S2S models, a list of attributes in an MR has to be linearized into a sequence of tokens (Konstas et al., 2017; Ferreira et al., 2017). Not all attributes have to appear for all inputs, and each attribute might have multi-token values, such as *area: city centre*. We use special start and stop tokens for each possible attribute to mark value boundaries; for example, an attribute *area: city centre* becomes *__start_area__ city centre __end_area__*. These fragments are concatenated into a single sequence to represent the original MR as an input sequence to our models. In this approach, no values are delexicalized, in contrast to Juraska et al. (2018) and others who delexicalize a subset of attributes. An alternative approach by Freitag and Roy (2018) treats the attribute type as an additional feature and learn embeddings for words and types separately.

4 Learning Content Selection

We extend the vanilla S2S system with methods that address the related problem of text summarization. In particular, we implement the pointer-generator network similar to that introduced by Nallapati et al. (2016) and See et al. (2017), which can generate content by copying tokens from an input during the generation process.

Copy Model The copy model introduces a binary variable z_t for each decoding step t that acts as a switch between copying from the source and generating words. We model the joint probability following the procedure described by Gulcehre et al. (2016) as

$$p(y_t, z_t|y_{[t-1]}, \mathbf{x}) = \sum_{z \in \{0,1\}} p(y_t, z_t = z|\mathbf{y}_{[t-1]}, \mathbf{x})$$

To calculate the switching probability $p(z_t|\mathbf{y}_{[t-1]}, \mathbf{x})$, let $v \in \mathbb{R}^{\text{d}_{\text{hid}}}$ be a trainable parameter. The hidden state of the decoder h_t is used to compute $p(z_t) = \sigma(h_t^T v)$ and decompose the joint distribution into two parts:

$$\begin{aligned} p(y_t|y_{[t-1]}, \mathbf{x}) = p(z_t = 1) \times p(y_t|z_t = 1) \\ + p(z_t = 0) \times p(y_t|z_t = 0), \end{aligned}$$

where every term is conditioned on $\mathbf{x}$ and $\mathbf{y}_{[t-1]}$. $p(y_t|z_t = 0)$ is the distribution generated by the previously described S2S model, and $p(y_t|z_t = 1)$ is a distribution over $\mathbf{x}$ that is computed using the same attention mechanism with separate parameters.

In our problem, all values in the MR's should occur in the generated text and are typically words that would not be generated by a language model. This allows us to use an assumption by Gulcehre et al. (2016) that every word that occurs in both source and target was copied, which avoids having to marginalize over z. Then, the log-likelihood of y_t and z_t is maximized during training. This approach has the further advantage that it can handle previously unseen input by learning to copy these words into the correct position.

Coverage and Length Penalty We observed that generated text using vanilla S2S models with and without copy mechanism commonly omits some of the values in their inputs. To mitigate this effect, we use two penalty terms during inference; a length and a coverage penalty. We are using a coverage penalty during inference only, opposed to Tu et al. (2016) who introduced a coverage penalty term into the attention of an S2S model for neural machine translation and See et al. (2017) who used the same idea for abstractive summarization. Instead, we use the penalty term cp defined by Wu et al. (2016) as

$$cp(\mathbf{x}, \mathbf{y}) = \beta \cdot \sum_{i=1}^{|\mathbf{x}|} \log(\min(\sum_{t=1}^{|\mathbf{y}|} a_i^t, 1.0)).$$

Here, β is a parameter to control the strength of the penalty. This penalty term increases when too many generated words attend to the same input. We typically do not want to repeat the name of the

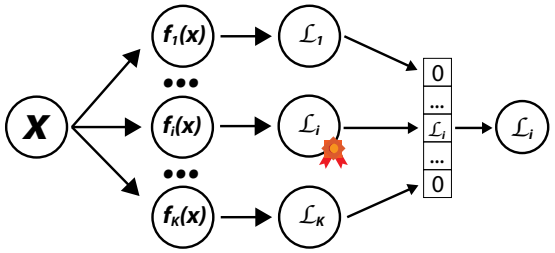

Figure 2: The multiple-choice loss for a single training example. $\mathcal{L}_i$ has the smallest loss and receives parameter updates.

restaurant or the type of food it serves. Thus, we only want to attend to the restaurant name once when we actually generate it. We also use the length penalty lp by Wu et al. (2016), defined as

$$lp(\mathbf{y}) = \frac{(5 + |\mathbf{y}|)^\alpha}{(5+1)^\alpha},$$

where α is a tunable parameter that controls how much the likelihoods of longer generated texts are discounted. The penalties are used to re-rank beams during the inference procedure such that the full score function s becomes

$$s(\mathbf{x}, \mathbf{y}, z) = \frac{\log p(\mathbf{y}, z|\mathbf{x})}{lp(\mathbf{y})} + cp(\mathbf{x}, \mathbf{y}).$$

A final inference time restriction of our model is the blocking of repeat sentence beginnings. Automatic metrics do not punish a strong parallelism between sentences, but repeat sentence beginnings interrupt the flow of a text and make it look unnatural. We found that since each model follows a strict latent template during generation, the generated text would often begin every sentence with the same words. Therefore, we encourage syntactic variation by pruning beams during beam search that start two sentences with the same bigram. Paulus et al. (2017) use similar restrictions for summarization by blocking repeated trigrams across the entire generated text. Since automated evaluation does not punish repeat sentences, we only enable this restriction when generating text for the human evaluation.

5 Learning Latent Sentence Templates

Each generated text follows a latent sentence template to describe the attributes in its MR. The model has to associate each attribute with its location in a sentence template. However, S2S models can learn wrong associations between inputs and targets with limited data, which was also shown by Ferreira et al. (2017). Additionally, consider that we may see the generated texts for similar inputs: *There is an expensive British Restaurant called the Eagle.* and *The Eagle is an expensive, British Restaurant.*. Both incorporate the same information but have a different structure. A model that is trained on both styles simultaneously might struggle to generate a single output sentence. To address this issue and to learn a set of diverse generation styles, we train a mixture of models where every sequence is still generated by a single model. The method aims to force each model to learn a distinct sentence template.

The mixture aims to split the training data between the models such that each model trains only on a subset of a data, and can learn a different template structure. Thus, one model does not have to fit all the underlying template structures simultaneously. Moreover, it implicitly removes outlier training examples from all but one part of the mixture. Let $f_1, \ldots, f_K$ be the K models in the mixture. These models can either be completely disjoint or share a subset of their parameters (e.g. the word embeddings, the encoder, or both encoder and decoder). Following Guzman-Rivera et al. (2012), we introduce an unobserved random variable $w \sim \text{Cat}(1/K)$ that assigns a weight to each model for each input. Let $p_\theta(\mathbf{y}|\mathbf{x}, w)$ denote the probability of an output $\mathbf{y}$ for an input $\mathbf{x}$ with a given segmentation w. The likelihood for each point is defined as a mixture of the individual likelihoods,

$$\begin{aligned}\log p(\mathbf{y}|\mathbf{x}) &= \log \sum_w p(\mathbf{y}, w|\mathbf{x}) \\ &= \log \sum_w p(w) \times p(\mathbf{y}|w, \mathbf{x}).\end{aligned}$$

By constraining w to assume either 0 or 1, the optimization problem over the whole dataset becomes a joint optimization of assignments of models to data points and parameters to models.

To maximize the target, Guzman-Rivera et al. (2012) propose a multiple-choice loss (MCL) to segment training data similar to a hard EM algorithm or k-Means clustering. With MCL, after each training epoch, each training point is assigned to the model that predicts it with the minimal loss. After this segmentation, each model is trained for a further epoch using only its assigned

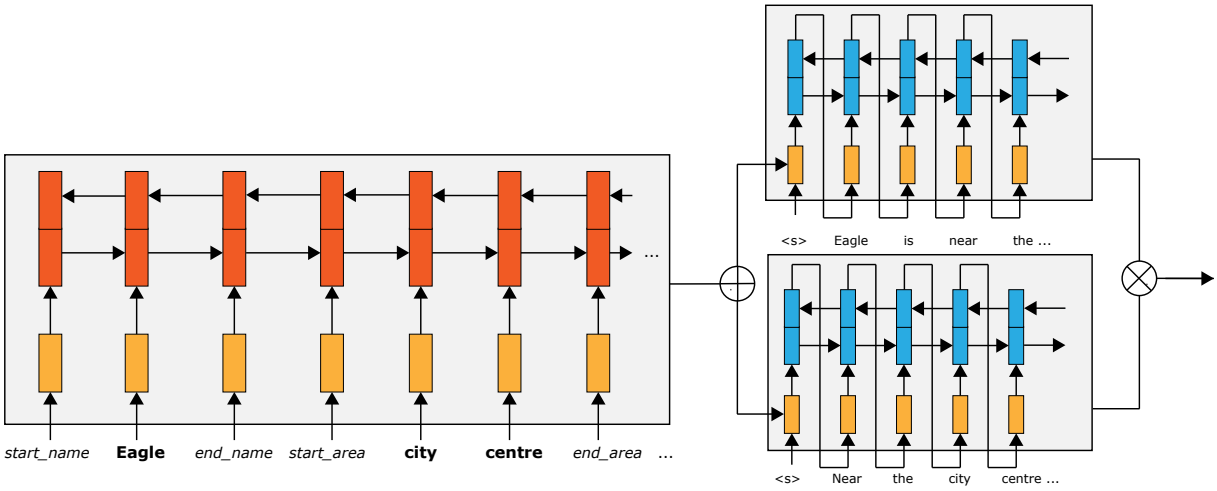

Figure 3: An illustration of the diverse ensembling method with $K = 2$ and a shared encoder. The encoder, shown on the left, reads the meaning representation and generates the contextual representations of the input tokens. The context is then used in parallel by the two separate decoders. Here, $\oplus$ represents the duplication of the input representation. The two decoders generate text independently from each other. Finally, only the decoder with the better generated text receives a parameter update. The exclusive choice is illustrated by the $\otimes$ operation.

data points. This process repeats until the point assignments converge. Related work by Kondadadi et al. (2013) has shown that models compute clusters of templates

Further work by Lee et al. (2016) reduce the computational overhead by introducing a stochastic MCL (sMCL) variant that does not require retraining. They compute the posterior over $p(w|\mathbf{x}, \mathbf{y})$ in the E-Step by choosing the best model for an example $\hat{k} = \text{argmax}_{k \in [K]} p_\theta(\mathbf{y}|\mathbf{x}, w_k = 1, w_{\neg k} = 0)$. Setting $w_{\hat{k}}$ to 1 and all other entries in w to 0 achieves a hard segmentation for this point. After this assignment, only the model $\hat{k}$ with the minimal negative log-likelihood is updated in the M-Step. A potential downside of this approach is the linear increase in complexity since a forward pass has to be repeated for each model.

We illustrate the process of a single forward-pass in Figure 2, in which a model f_i has the smallest loss $\mathcal{L}_\rangle$ and is thus updated. Figure 3 demonstrates an example with $K = 2$ in which the two models generate text according to two different sentence layouts. We find that averaging predictions of multiple models during inference, a technique commonly used with traditional ensembling approaches, does not lead to increased performance. We further confirm findings by Lee et al. (2017) who state that these models overestimate their confidence when generating text. Since it is our goal to train a model that learns the best underlying template instead of generating diverse predictions, we instead generate text using only the model in the ensemble with the best perplexity on the validation set.

Attribute	Value
area	city centre, riverside, ...
customerRating	1 out of 5, average, ...
eatType	coffee shop, restaurant, ...
familyFriendly	yes / no
food	Chinese, English, ...
name	Wildwood, The Wrestlers, ...
near	Café Sicilia, Clare Hall, ...
priceRange	less than £20, cheap, ...

Table 1: A list of all possible attributes and some example values for the E2E NLG dataset.

6 Experiments

We apply our method to the crowd-sourced E2E NLG dataset of Novikova et al. (2017) that comprises 50,000 examples of dialogue act-based MRs and reference pairs in the restaurant domain. Each input is a meaning representation of on average 5.43 attribute-value pairs, and the target a corresponding natural language utterance. A list of possible attributes is shown in Table 1. The dataset is split into 76% training, and 9% validation, and 15% test data. The validation and test data are

#	Setup	BLEU	NIST	METEOR	ROUGE	CIDEr
	TGEN (Dušek and Jurčíček, 2016)	69.3	8.47	47.0	72.6	2.39
	Ensemble with Slot Filling (Juraska et al., 2018)	69.3	8.41	43.8	70.1	/
	Hierarchical Decoding (Su et al., 2018)	44.1	/	/	53.8	/
	S2S with Slot Embeddings (Freitag and Roy, 2018)	72.7	8.3	/	75.1	/
(1)	*mlp*	70.6	8.35	47.3	73.8	2.38
(2)	*dot*	71.1	8.43	47.4	73.7	2.35
(3)	*mlp*, copy	71.4	8.44	47.0	74.1	2.43
(4)	*dot*, copy	69.8	8.20	47.8	74.3	2.51
(5)	*mlp*, $K = 2$	72.6	8.70	48.5	74.8	2.52
(6)	*dot*, $K = 2$	73.3	8.68	**49.2**	**76.3**	2.61
(7)	*mlp*, copy, $K = 2$	73.6	8.74	48.5	75.5	**2.62**
(8)	*dot*, copy, $K = 2$	**74.3**	**8.76**	48.1	75.3	2.55
(9)	Transformer	69.0	8.22	47.8	74.9	2.45
(10)	Transformer, $K = 2$	73.7	8.75	48.9	**76.3**	2.56

Table 2: Results of different S2S approaches and published baseline models on the E2E NLG validation set. The second section shows models without diverse ensembling, the third section with it. The fourth section shows results of the Transformer model. / indicates that numbers were not reported.

multi-reference; the validation set has on average 8.1 references for each MR. A separate test set with previously unseen combinations of attributes contains 630 MR's and its references are unseen and used for evaluation in the E2E NLG challenge.

For all LSTM-based S2S models, we use a two-layer bidirectional LSTM encoder, and hidden and embedding sizes of 750. During training, we apply dropout with probability 0.2 and train models with Adam (Kingma and Ba, 2014) and an initial learning rate of 0.002. We evaluate both *mlp* and *dot* attention types. The Transformer model has 4 layers with hidden and embedding sizes 512. We use the training rate schedule described by Vaswani et al. (2017), using Adam and a maximum learning rate of 0.1 after 2,000 warm-up steps. The diverse ensembling technique is applied to all approaches, pre-training all models for 4 epochs and then activating the sMCL loss. All models are implemented in OpenNMT-py (Klein et al., 2017)[2]. The parameters were found by grid search starting from the parameters used in the TGEN model by Dušek and Jurčíček (2016). Unless stated otherwise, models do not block repeat sentence beginnings, since it results in worse performance in automated metrics. We show results on the multi-reference validation and the blind test sets for the five metrics BLEU (Papineni et al., 2002), NIST (Doddington, 2002), METEOR (Denkowski and Lavie, 2014), ROUGE (Lin, 2004), and CIDEr (Vedantam et al., 2015).

[2]Code and documentation can be found at https://github.com/sebastianGehrmann/diverse_ensembling

7 Results

7.1 Results on the Validation Set

Table 2 shows the results of different models on the validation set. During inference, we set the length penalty parameter α to 0.4, the coverage penalty parameter β to 0.1, and use beam search with a beam size of 10. Our models outperform all shown baselines, which represent all published results on this dataset to date. Except for the copy-only condition, the data-efficient *dot* outperforms *mlp*. Both copy-attention and diverse ensembling increase performance, and combining the two methods yields the highest BLEU and NIST scores across all conditions. The Transformer performs similarly to the vanilla S2S models, with a lower BLEU but higher ROUGE score. Diverse ensembling also increases the performance with the Transformer model, leading to the highest ROUGE score across all model configurations. Table 3 shows generated text from different models. We can observe that the model without copy attention omits the rating, and without ensem-

bling, the sentence structure repeats and thus looks unnatural. With ensembling, both models produce sensible output with different sentence layouts. We note that often, only the better of the two models in the ensemble produces output better than the baselines. We further analyze how many attributes are omitted by the systems in Section 7.3.

To analyze the effect of length and coverage penalties, we show the average relative change across all metrics for model (8) while varying α and β in Figure 4. Both penalties increase average performance slightly, with an average increase of the scores by up to 0.82%. We find that recall-based metrics increase while the precision-based metrics decrease when applying the penalty, which can be explained by an increase in the average length of the generated text by up to 2.4 words. Results for ensembling variations of model (8) are shown in Table 4. While increasing K can lead to better template representations, every individual model will be trained on fewer data points. This can result in an increased generalization error. Therefore, we evaluate updating the top 2 models during the M-step and setting K=3. While increasing K from 2 to 3 does not show a major increase in performance when updating only one model, the K=3 approach slightly outperforms the K=2 one with the top 2 updates.

Having the K models model completely disjoint data sets and use a disjoint set of parameters could be too strong of a separation. Therefore, we investigate the effect of sharing a subset of the parameters between individual models. Our results in rows (5)-(7) of Table 4 show only a minor improvement in recall-based approaches when sharing the word embeddings between models but at the cost of a much lower BLEU and NIST score. Sharing more parameters further harms the model's performance.

7.2 Results on the Blind Test Set

We next report results of experiments on a held-out test set, conducted by the E2E NLG challenge organizers (Dušek et al., 2018), shown in Table 5. The results show the validity of the approach, as our systems outperform competing systems in these; ranking first in ROUGE and CIDEr and sharing the first rank in METEOR. The first row of the table shows the results with blocked repeat sentence beginnings. While this modification leads to slightly reduced scores on the automated

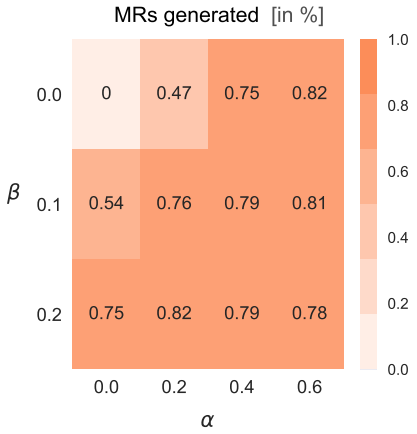

Figure 4: Relative change of performance averaged over all five metrics when varying inference parameters for model (8). Length penalty parameter α controls length, and coverage penalty parameter β penalizes source values with no attention.

MR	name: Wildwood; eatType: coffee shop; food: English; priceRange: moderate; customerRating: 3 out of 5; near: Ranch
(1)	Wildwood is a coffee shop providing English food in the moderate price range. It is located near Ranch.
(4)	Wildwood is a coffee shop providing English food in the moderate price range. It is near Ranch. Its customer rating is 3 out of 5.
(8).1	Wildwood is a moderately priced English coffee shop near Ranch. It has a customer rating of 3 out of 5.
(8).2	Wildwood is an English coffee shop near Ranch. It has a moderate price range and a customer rating of 3 out of 5.

Table 3: Examples of generated text by different systems for the same MR, shown in the first line. Numbers correspond to model configurations in Table 2.

metrics, it makes the text look more natural, and we thus use this output in the human evaluation.

The human evaluation compared the output to 19 other systems. For a single meaning representation, crowd workers were asked to rank output from five systems at a time. Separate ranks were collected for the *quality* and *naturalness* of the generations. The ranks for quality aim to reflect the grammatical correctness, fluency, and adequacy of the texts with respect to the structured input. In order to gather ranks for the naturalness, generations were shown without the meaning representation and rated based on how likely an utterance could have been produced by a na-

#	Setup	BLEU	NIST	METEOR	ROUGE	CIDEr
(1)	$K = 1$	69.8	8.20	47.8	74.3	2.51
(2)	$K = 2$	**74.3**	8.76	48.1	75.3	2.55
(3)	$K = 3$	73.6	8.73	48.8	75.5	**2.64**
(4)	$K = 3$, top 2	74.2	**8.81**	**48.6**	**76.1**	2.56
(5)	$K = 2$, share embedding	73.1	8.61	48.6	75.4	2.58
(6)	$K = 2$, share encoder	72.2	8.56	47.8	74.4	2.50
(7)	$K = 2$, share encoder + decoder	72.4	8.43	47.3	74.6	2.50

Table 4: Variants of diverse ensembling. The top section shows results of varying the number of models in a diverse ensemble on the validation set. The bottom section shows results with different numbers of shared parameters between two models in a diverse ensemble. All results are generated with setup (8) from Table 2.

Setup	BLEU	NIST	METEOR	ROUGE	CIDEr
TGEN (Dušek and Jurčíček, 2016)	65.9	8.61	44.8	68.5	2.23
Slot Filling (Juraska et al., 2018)	66.2	8.31	44.5	67.7	2.26
dot, $K = 3$, top 2, block repeats	65.0	8.53	43.9	68.7	2.09
dot, $K = 3$, top 2	65.8	8.57 (8)	44.1	68.9 (9)	2.11
Transformer, $K = 2$	66.2 (8)	8.60 (7)	**45.7 (1)**	70.4 (3)	**2.34 (1)**
dot, copy, $K = 2$	67.4 (3)	8.61 (6)	45.2 (4)	**70.8 (1)**	2.31 (3)

Table 5: The results of our model on the blind E2E NLG test set. Notable rankings within the 60 submitted systems are shown in parentheses. Systems by Freitag and Roy (2018) and Su et al. (2018) were not evaluated on this set.

tive speaker. The results were then analyzed using the TrueSkill algorithm by Sakaguchi et al. (2014). The algorithm produced 5 clusters of systems for both quality and naturalness. Within clusters, no statistically significant difference between systems can be found. In both evaluations, our main system was placed in the second best cluster. One difference between our and the system ranked first in quality by Juraska et al. (2018) is that our model frequently fails to generate text about inputs despite the coverage penalty.

7.3 Which Attributes do the Models Generate?

Vanilla S2S models frequently miss to include attributes of an MR, even though almost all the training examples use all of them. While Juraska et al. (2018) adds an explicit penalty for each attribute that is not part of a generated text, we aim to implicitly reduce this number with the coverage penalty. To investigate the effectiveness of the model extensions, we apply a heuristic that matches an input with exact word matches in the generated text. This provides a lower bound to the number of generated attributes since paraphrases are not captured. We omit the *familyFriendly* category from this figure since it does not work with this heuristic.

In Figure 5 (a) we show the cumulative effect of model extensions on generated attributes across all categories. Copy attention and the coverage penalty have a major effect on this number, while the ensembling only slightly improves it. In Figure 5 (b), we show a breakdown of the generated attributes per category. The base model struggles with *area*, *price range*, and *customer rating*. Price range and customer rating are frequently paraphrased, for example by stating that a restaurant with a 4 out of 5 rating has a good rating, while the area cannot be rephrased. While customer rating is one of the most prevalent attributes in the data set, the other two are more uncommon. The full model improves across almost all of the categories but also has problems with the price range. The only category in which it performs worse is the name category, which could be a side effect of the particular split of the data that the model learned. Despite the decrease in mistakenly omit-

Figure 5: (a): The figure shows a lower bound on the percentage of all attributes the model is generating for each model type. The base model is missing almost 40% of all inputs. (b) The figure shows a breakdown per attribute how many the model is generating compared to the reference.

ted attributes, the model still misses up to 20% of attributes. We hope to address this issue in future work by explicitly modeling the underlying slots and penalizing models when they ignore them.

8 Conclusion

In this paper, we have shown three contributions toward end-to-end models for data-to-text problems. We surveyed existing S2S modeling methods and extensions to improve content selection in the NLG problem. We further showed that applying diverse ensembling to model different underlying generation styles in the data can lead to a more robust learning process for noisy data. Finally, an empirical evaluation of the investigated methods showed that they lead to improvements across multiple automatic evaluation metrics. In future work, we aim to extend the shown methods to address generation from more complex inputs, and for challenging domains such as data-to-document generation.

9 Acknowledgements

We thank the three anonymous reviewers for their valuable feedback. This work was supported by a Samsung Research Award.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pages 376–380.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145. Morgan Kaufmann Publishers Inc.

Ondřej Dušek and Filip Jurčíček. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. *arXiv preprint arXiv:1606.05491*.

Ondrej Dušek, Jekaterina Novikova, and Verena Rieser. 2018. Findings of the E2E NLG challenge. In *(in prep.)*.

Thiago Castro Ferreira, Iacer Calixto, Sander Wubben, and Emiel Krahmer. 2017. Linguistic realisation as machine translation: Comparing different mt models for amr-to-text generation. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 1–10.

Markus Freitag and Scott Roy. 2018. Unsupervised natural language generation with denoising autoencoders. *arXiv preprint arXiv:1804.07899*.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. *arXiv preprint arXiv:1603.08148*.

Abner Guzman-Rivera, Dhruv Batra, and Pushmeet Kohli. 2012. Multiple choice learning: Learning to produce multiple structured outputs. In *Advances in Neural Information Processing Systems*, pages 1799–1807.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Juraj Juraska, Panagiotis Karagiannis, Kevin Bowden, and Marilyn Walker. 2018. A deep ensemble model

with slot alignment for sequence-to-sequence natural language generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 152–162.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*.

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 Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1406–1415.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 146–157.

Gerasimos Lampouras and Andreas Vlachos. 2016. Imitation learning for language generation from unaligned data. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1101–1112. The COLING 2016 Organizing Committee.

Kimin Lee, Changho Hwang, KyoungSoo Park, and Jinwoo Shin. 2017. Confident multiple choice learning. *arXiv preprint arXiv:1706.03475*.

Stefan Lee, Senthil Purushwalkam Shiva Prakash, Michael Cogswell, Viresh Ranjan, David Crandall, and Dhruv Batra. 2016. Stochastic multiple choice learning for training diverse deep ensembles. In *Advances in Neural Information Processing Systems*, pages 2119–2127.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8. Barcelona, Spain.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

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

Hongyuan Mei, Mohit Bansal, and Matthew R Walter. 2015. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. *arXiv preprint arXiv:1509.00838*.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Jekaterina Novikova, Ondrej Dušek, and Verena Rieser. 2017. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Saarbrücken, Germany. ArXiv:1706.09254.

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*, pages 27–32. Association for Computational Linguistics.

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 association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Xiaochang Peng, Chuan Wang, Daniel Gildea, and Nianwen Xue. 2017. Addressing the data sparsity issue in neural amr parsing. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 366–375.

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2014. Efficient elicitation of annotations for human evaluation of machine translation. In *WMT@ ACL*, pages 1–11.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*.

Amanda Stent, Rashmi Prasad, and Marilyn Walker. 2004. Trainable sentence planning for complex information presentation in spoken dialog systems. In *Proceedings of the 42nd annual meeting on association for computational linguistics*, page 79. Association for Computational Linguistics.

Shang-Yu Su, Kai-Ling Lo, Yi Ting Yeh, and Yun-Nung Chen. 2018. Natural language generation by hierarchical decoding with linguistic patterns. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 61–66.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. *arXiv preprint arXiv:1601.04811*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Marilyn A Walker, Amanda Stent, François Mairesse, and Rashmi Prasad. 2007. Individual and domain adaptation in sentence planning for dialogue. *Journal of Artificial Intelligence Research*, 30:413–456.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. *arXiv preprint arXiv:1508.01745*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.