Handling Rare Items in Data-to-Text Generation

Anastasia Shimorina
Lorraine University / LORIA, Nancy, France
anastasia.shimorina@loria.fr

Claire Gardent
CNRS / LORIA, Nancy, France
claire.gardent@loria.fr

Abstract

Neural approaches to data-to-text generation generally handle rare input items using either delexicalisation or the copy mechanism. We investigate the relative impact of these two approaches on two datasets (E2E and WebNLG) and using two evaluation settings. We show (i) that rare items strongly impact performance; (ii) that combining delexicalisation and copying yields the strongest improvment; (iii) that copying underperforms for rare and unseen items and (iv) that the impact of these two mechanisms greatly varies depending on how the dataset is constructed and on how it is split into train, dev and test[1].

1 Introduction

The input to data-to-text generation often contains rare items, i.e. low frequency items such as names, locations and dates. This makes it difficult for neural models to learn their verbalisation. To address these issues, neural approaches to data-to-text generation typically resort either to delexicalisation (Wen et al., 2015; Dušek and Jurcicek, 2015; Trisedya et al., 2018; Chen et al., 2018) or to a copy mechanism (Chen, 2018; Elder et al., 2018; Gehrmann et al., 2018). Character-based encodings (Agarwal and Dymetman, 2017; Deriu and Cieliebak, 2018) and byte pair encodings have also been used (Elder, 2017; Zhang et al., 2018). When applying a character-based approach within a standard sequence-to-sequence model to the WebNLG and E2E datasets, the results were low. Hence we chose not to discuss them.

[1]All the data, scripts and evaluation results used in this study can be found at https://gitlab.com/shimorina/inlg-2018.

When using delexicalisation, the data is preprocessed to replace rare items with placeholders and the generated text is post-processed to replace these placeholders with appropriate values based on a mapping between placeholders and initial values built during preprocessing. While this method is often used, it has several drawbacks. It requires an additional pre- and post-processing step. These processing steps must be re-implemented for each new data-to-text application. The matching procedure needed to correctly match a rare input item (e.g., Barack Obama) with the corresponding part in the output text (e.g., the former President of the United States) may be quite complex which in turn may result in incorrect or incomplete delexicalisations. In contrast, the copy mechanisms standardly used in neural approaches to summarisation (See et al., 2017; Gu et al., 2016; Cheng and Lapata, 2016), paraphrasing (Cao et al., 2017), answer generation (He et al., 2017) and data-to-text generation (Gehrmann et al., 2018; Chen et al., 2018) is a generic technique which is easy to integrate in the encoder-decoder framework and can be used independently of the particular domain and application.

In this paper, we investigate the impact of copying and delexicalisation on the quality of generated texts using two sequence-to-sequence models with attention: one using the copy and coverage mechanism of See et al. (2017), the other using delexicalisation. We evaluate their respective output on two data-to-text datasets, namely the E2E (Novikova et al., 2017) and the WebNLG (Gardent et al., 2017a) datasets. We also compare the two methods in two different settings: the original train/dev/test partition produced by the E2E and by the WebNLG challenge *vs.* a more constrained train/dev/test split which aims to further minimise the amount of redundancy between train, dev and test data. This latter experimental setting is in-

MR	Reference
Original: name[The Cricketers], eatType[coffee shop], food[Chinese], customer rating[average], familyFriendly[no], near[The Portland Arms] **Delexicalised**: name[NAME], eatType[coffee shop], food[Chinese], customer rating[average], familyFriendly[no], near[NEAR]	**Original**: The Cricketers is a coffee shop that also has Chinese food, located near The Portland Arms. It is not family friendly, and has an average customer rating. **Delexicalised**: NAME is a coffee shop that also has Chinese food, located near NEAR. It is not family friendly, and has an average customer rating.
Original: (Bakewell pudding – region – Derbyshire Dales), (Bakewell pudding – dishVariation – Bakewell tart), (Bakewell pudding – servingTemperature – Warm or cold), (Bakewell pudding – course – Dessert), (Bakewell pudding – mainIngredients – Ground almond, jam, butter, eggs) **Delexicalised**: (FOOD – region – REGION), (FOOD – dishVariation – DISHVARIATION), (FOOD – servingTemperature – SERVINGTEMPERATURE), (FOOD – course – DESSERT), (FOOD – mainIngredients – MAININGREDIENTS)	**Original**: Bakewell pudding, also called bakewell tart, originates from the Derbyshire Dales. Classified as a dessert which can be served warm or cold, its main ingredients are ground almond, jam, butter and eggs. **Delexicalised**: FOOD, also called DISHVARIATION, originates from the REGION. Classified as a COURSE which can be served SERVINGTEMPERATURE, its main ingredients are MAININGREDIENTS.

Table 1: Entry examples of the E2E (first row) and WebNLG (second row) datasets with and without delexicalisation.

spired by a recent paper by Aharoni and Goldberg (2018), which shows that the train/dev/test split may have a strong impact on how much the model learns to generalise and how much it memorises.

Our study brings to light the following points.

- Rare items strongly impact the performance of Data-to-Text generation.
- Combining delexicalisation and copying yields the strongest improvments.
- Copying underperforms for items not or rarely seen in the training data.
- The content (e.g., distribution and number of named entities) and the partitioning (constraints on the test set) of the training data strongly affect the impact of both copying and delexicalisation.

2 Experiments

2.1 Datasets

Two recently released corpora for data-to-text generation served as experimental datasets for our study, namely the E2E (Novikova et al., 2017) and the WebNLG (Gardent et al., 2017a) datasets.

In the E2E dataset, the input to generation is a dialogue act consisting of three to eight slot-value pairs describing a restaurant, while the output is a restaurant recommendation verbalising this input. Table 1 shows an example with an input consisting of six slot-value pairs. In average, each input is associated with 8.1 references. The number of possible values for each slot ranges from two (binary slots) to 34 (restaurant name). Tables 2 and 3 summarises the statistics of the E2E dataset.

In WebNLG, the aim is to verbalise a set of RDF (Resource Description Framework) triples describing entities of different categories. An RDF triple is of the form *(subject, property, object)* where *subject* and *object* denotes entities or values and *property* denotes a binary relation holding between *subject* and *object*. The inputs consist of sets of (one to seven) triples and the entities belong to fifteen distinct DBpedia categories[2].

Both dataset releases gave rise to a shared task in NLG in 2017[3]. Note though that for WebNLG, the present study relies on the final release data (version 2)[4], which is a larger dataset than that used for the WebNLG Challenge 2017.

2.2 Delexicalising Datasets

We derive delexicalised datasets from the original E2E and WebNLG datasets as follows.

For each dataset, we replicated the delexicalisation procedure which was applied to the baseline systems developed for the E2E (Novikova et al., 2017) and for the WebNLG challenge (Gardent et al., 2017b) respectively[5]. As shown in Table 1, both input data and output text were delexicalised. In E2E, only the *name* and *near* slots were

[2]`http://wiki.dbpedia.org/dbpedia-dataset-version-2015-10`

[3]`http://www.macs.hw.ac.uk/InteractionLab/E2E/`, `http://webnlg.loria.fr/pages/results.html`

[4]available at `https://gitlab.com/shimorina/webnlg-dataset`

[5]For the full details of these delexicalisation procedures, see (Novikova et al., 2017; Gardent et al., 2017b) and the webpages of the two challenges: `http://www.macs.hw.ac.uk/InteractionLab/E2E/`, `http://webnlg.loria.fr/pages/results.html`

Attribute	Value Range	Example
area	2	city centre, riverside
customer rating	6	3 out of 5, low, high
eatType	3	restaurant, coffee shop, pub
familyFriendly	2	no, yes
food	7	English, Chinese, Fast food
name	34	The Plough, Alimentum, Zizzi
near	19	Café Sicilia, Crowne Plaza Hotel
priceRange	6	more than £30, cheap, moderate

	Count	Example
Properties	373	dateOfBirth, genre
Subjects	732	Buzz Aldrin
Objects	2,916	1932-03-15, jazz

Table 2: Statistics on attribute values in E2E (left) and on RDF-triple constituents in WebNLG (right).

	E2E Dataset				WebNLG Dataset			
	Unconstrained		*Constrained*		*Unconstrained*		*Constrained*	
	Instances	MRs	Instances	MRs	Instances	MRs	Instances	MRs
train	40,868	4,862	40,826	4,877	34,352	12,876	34,536	12,895
dev	4,521	547	3,946	547	4,316	1,619	4,217	1,594
test	4,577	630	5,194	615	4,224	1,600	4,148	1,606

Table 3: Training/development/test sets statistics in E2E and WebNLG in original (unconstrained) and constrained splits. *Instances* count is a number of (data, text) pairs; *MRs* count is a number of unique data inputs.

delexicalised (because contrary to the other slots, they have a large number of distinct values). In WebNLG, delexicalisation was done on the subjects and objects of RDF triples.

While delexicalisation was flawless in E2E, WebNLG data poses additional challenges as the subject and object values in the input do not necessarily match the corresponding text fragment in the output. As a result, not all subjects and objects were delexicalised.

In the delexicalised E2E corpus, placeholders constitute 5.7% of all tokens, while they reach 15.7% in the WebNLG data.

2.3 The Copy Mechanism

The copy mechanism is widely used in text production approaches where it is relevant for handling rare input but also, for instance, in text summarisation, for copying input into the output. Thus, Cao et al. (2017) uses a copy mechanism to generate paraphrases, Gu et al. (2016), Cheng and Lapata (2016) for text summarisation and He et al. (2017) for answer generation.

Here we use the copy mechanism introduced by See et al. (2017). The decoder uses an extended vocabulary which consists of a predefined target vocabulary P_{vocab} which is dynamically extended at inference time with the tokens contained in the input. At each time step during decoding, the model then decides whether to copy from the input or to generate from the target vocabulary using a probability distribution over the extended vocabulary which is computed based on a generation probability (sampling from the target vocabulary) and on the attention distribution (sampling from the input).

The attention distribution a_t is calculated as in (Bahdanau et al., 2015):

$$e_i^t = v^T tanh(W_h h_i + W_s s_t + b_{attn})$$
$$a^t = softmax(e^t)$$

with v, W_h, W_s and b_{attn} parameters to be learned, s_t is the decoder state and h_i is a variable ranging over the encoder hidden states.

The generation probability p_{gen} is then defined as:

$$p_{gen} = \sigma(W_h^T.h_t + W_s^T.s_t + W_x^T.x_t + b_{ptr})$$

where W_h, W_s, W_x, b_{ptr} are parameters to be learned, x_t is the decoder input and h_t is the context vector produced by the attention mechanism as the weighted sum $\sum_1^E a_i^t h_i$ of the encoder states (with N the number of encoder states).

Finally, the probability distribution over the extended vocabulary is defined as:

$$P(w) = p_{gen}P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=wa_i^t} a_i^t$$

2.4 Constraining Datasets

The train/dev/test split is often constrained to ensure that there is no overlap in terms of *input* between the training, the development and the test set. As Aharoni and Goldberg (2018) recently showed however, this may result in a setup where certain *input fragments* (in that case, subject and object entities present in the input set of RDF triples) are present so often in the test set that models built on this standard split, overfit and memorise rather than learn. Thus, in the split-and-rephrase application they studied, Aharoni and Goldberg (2018) observed that, given some input containing the entity e and some set of facts $T(e)$ about this entity, the model will regularly output a text which mentions e but is unrelated to the set of facts $T(e)$. That is, instead of learning to generate text from data, the model learns to associate a text with an entity.

To better assess the impact of delexicalisation and copying on the output of data-to-text generation models, we therefore consider two ways of partitioning the corpus into train, dev and test: the traditional way (*Unconstrained*) where the overlapping constraint applies to entire inputs (i.e., sets of RDF triples in WebNLG and dialogue moves in E2E) and a more challenging split (*Constrained*) where the no-overlap constraint applies to input fragments (i.e., RDF triples in WebNLG and slot-values in E2E). Table 3 shows the statistics for both splits for each dataset.

Unconstrained The unconstrained split is the original split provided by the challenge organisers.

The E2E dataset was split into training, validation and test sets following a 76.5/8.5/15 ratio. It was ensured that the input were distinct for all three sets and that a similar distribution of input and reference text lengths was kept. We found 1,430 identical (data, text) pairs in the original E2E data. They were deleted for the subsequent experiments.

In WebNLG, the original split follows an 80/10/10 ratio. As with the E2E dataset, there is a null intersection in terms of input between train, dev and test. In addition, sets of triples of different sizes and sets of triples of different categories were proportionally distributed between training, dev and test sets.

Constrained We consider a second partitioning where we aim to minimise the overlap between train, dev and test in terms of input fragments.

As shown in Table 2, in the E2E dataset, most of the slots have under eight possible values. As these few values appear with a large number of distinct slot-value combinations (49,966 input-text instances), they are unlikely to trigger fact memorisation. We therefore focus on those slots which have a higher number of values and restrict the test set using restaurant names, a slot with 34 possible values. Four restaurant names were selected to occur only in the test data and to support a distribution of inputs types and text length similar to that of the original train/dev/test (cf. Table 3).

Nonetheless, it is worth noting that the E2E dataset was constructed in such a way that a specific restaurant name may have mutually exclusive values in different inputs, such as *low customer rating* and *high customer rating*. This might result in weak association between restaurant names and specific inputs and therefore, in little risk of memorising facts related to a specific restaurant name. As we shall see in Section 3, this intuition is confirmed by the results which show little differences, for the E2E data, in terms of both automatic and human-based metrics between the *Constrained* and the *Unconstrained* setting. Note also that since the E2E *Constrained* split is defined with respect to a slot value (restaurant names) which is delexicalised, the constrained *vs.* unconstrained split distinction loses its impact in the delexicalised setting.

For the WebNLG dataset, the constraint on the train/dev/test partition is in terms of triples. In addition to the exclusion from the test set all inputs (set of RDF triples) which occur in either the dev or the train set, we require that no RDF triple occurs in two of these sets. Let $t = (s, p, o)$ be an RDF-triple, with p a property and s, o subject and object RDF resources. In the constrained dataset, if, t is in the test set, then t may not be in either the dev or the training set but variants such as (s', p, o), (s, p, o'), (s', p, o') or (s, p', o) may (with $s \neq s'$, $p \neq p'$ and $o \neq o'$). In this way, models can be trained which must learn to verbalise properties independently of their arguments. Again, care was taken to keep the distribution in terms of input length similar to that of the original split (cf. Table 3).

2.5 Model Parameters

We trained two types of models: a standard sequence-to-sequence model and the same model augmented with a copy and coverage mechanism (denoted as *cc* in the tables). For the standard sequence-to-sequence model, we made use

E2E	name[Cocum], eatType[pub], customer rating[high], near[Burger King]
prediction	Cotto is a family-friendly pub with a high customer rating.
annotation	Cotto {*wrong*}, family-friendly {*added*}, pub {*right*}, high customer rating {*right*}, near Burger King {*missed*}
WebNLG	A Wizard of Mars – author – Diane Duane
prediction	A Wizard of Mars was written in the United States in 1995.
annotation	A Wizard of Mars {*right*}, was written {*right*}, in the United States {*wrong*}, in 1995 {*added*}.

Table 4: Manual annotation of text predictions for E2E and WebNLG data. Annotations are between curly braces.

of an LSTM encoder-decoder model with attention (Luong et al., 2015) from the OpenNMT-py toolkit[6], a PyTorch port of OpenNMT (Klein et al., 2017). The default parameters of OpenNMT-py were used for training and decoding. The encoder and decoder both have two layers. Models were trained for 13 epochs, with a mini-batch size of 64, a dropout rate of 0.3, and a word embedding size of 500. They were optimised with SGD with a starting learning rate of 1.0.

Data was not lowercased, nor was it truncated (the maximal sequence length was used in the source and target).

Special options available in OpenNMT-py were used to augment the standard model with the copy and the coverage mechanisms. The OpenNMT-py implementation of training additional copy and coverage attention layers follows See et al. (2017).

2.6 Evaluation

Automatic Evaluation Systems were evaluated using four automatic corpus-based metrics: BLEU (Papineni et al., 2002), NIST (Doddington, 2002), METEOR (Denkowski and Lavie, 2014), ROUGE$_L$ (Lin, 2004). We made use of the scripts used for the E2E Challenge evaluation[7]. The first three metrics were originally developed for machine translation, the last one for summarisation. Roughly speaking, BLEU calculates the n-gram precision; NIST is based on BLEU, but adds more weight to rarer n-grams; METEOR computes the harmonic mean of precision and recall, featuring also stem and synonymy matching; ROUGE$_L$ calculates recall for common longest subsequences in a reference and candidate text. Given our task—handling rare items (or named entities in the corpora in question)—we also applied the slot-error rate (SER) to evaluate outputs which seems to be more suitable for evaluating the presence of named entities. SER was calculated by exact matching slot values in the candidate texts,

$$SER = \frac{S + D + I}{N},$$

where S is a number of substitutions, D is a number of deletions, I is a number of insertions, and N is a total number of slots in the reference. The resulting SER is an average of SER for each prediction. While computing SER for the dialogue slot-based E2E corpus is straightforward (the binary slot *familyFriendly* was excluded), it results in some noise for WebNLG where subjects and objects are numerous (3,648 vs. 79 values in E2E) and where they were rephrased in references (cf. also Section 2.2).

Manual Annotation To allow comparisons between constrained and unconstrained settings, we intersected inputs of constrained and unconstrained test sets and gathered corresponding predictions from them for all the models. The intersection between the two test sets has 40 inputs in the E2E corpus and 153 in WebNLG. For E2E, we manually evaluated all 40 predictions available for each system (constrained and unconstrained); for WebNLG, we chose 44 predictions ensuring the presence of different sizes and categories. By manually assessing outputs for the same inputs for all the systems, contrasts between constrained and unconstrained settings are better highlighted.

Manual inspection of outputs revealed that most of generated predictions did not encounter issues with grammar or fluency. For this reason, we chose to focus on semantic adequacy of predicted texts. The evaluation was done by one human judge. After the evaluation was finished, the human annotator confirmed that, except for one system (see Section 3), all system outputs demonstrated fluent and grammatical English sentences.

Once presented with an input and a corresponding prediction text, a human judge was asked to evaluate semantic information present in the prediction. A minimal unit of analysis was a slot-value pair in E2E and an RDF triple element (sub-

[6]https://github.com/OpenNMT/OpenNMT-py
[7]https://github.com/tuetschek/e2e-metrics

	Unconstrained				*Constrained*			
	∅	*C*	*D*	*D*+*C*	∅	*C*	*D*	*D*+*C*
BLEU	0.56	**0.68**	0.67	**0.68**	0.52	0.57	**0.72**	**0.72**
NIST	7.54	8.67	8.60	**8.74**	7.12	7.67	**8.93**	8.90
METEOR	0.38	**0.46**	0.45	**0.46**	0.39	0.41	**0.47**	**0.47**
ROUGE$_L$	0.62	**0.71**	0.70	0.70	0.58	0.62	**0.74**	**0.74**
SER	26.07%	7.25%	**4.08%**	4.56%	29.6%	17.09%	5.18%	**4.2%**
right	81.72%	95.7%	96.24%	**96.77%**	72.04%	82.8%	**95.7%**	94.09%
wrong	16.13%	0%	0%	0%	27.96%	16.13%	0%	0%
missed	2.15%	4.3%	3.76%	3.23%	0%	1.08%	4.3%	5.91%
added	6.45%	0%	5.38%	2.15%	0%	4.84%	2.15%	0%

Table 5: E2E dataset (*D*: Delexicalisation, *D*+*C*: delexicalisation and copying, *C*: copy and coverage, ∅: Neither copy nor delexicalsiation. The upper half of the table presents automatic evaluation results; the lower half—human evaluation results. Best scores are in bold.

ject, object, or property) in WebNLG. For each semantic unit, the judge indicated if it was rendered correctly (*right*) or incorrectly (*wrong*) in the text. If the unit was missing, it was noted as *missed*; new semantic content, not present in the source input, was labelled as *added*. Then, the number of each type of annotations was calculated for each input and converted to percentage with respect to the number of slot-value pairs (E2E) or number of triple constituents (WebNLG). Given the E2E example in Table 4, statistics about the example is the following: right: 2, wrong: 1, added: 1, missed: 1 (*near[Burger King]* was omitted). Total number of slots being 4, the performance in the percentage is then right: 50%, wrong: 25%, added: 25%, missed: 25%.

WebNLG example annotations were done taking into account the three parts of a triple. If a property was not translated correctly, we considered that a model missed out that information. While a subject or object was not rendered correctly, they were annotated as wrong. All the semantic information beyond the size of initial set of triples was evaluated as added. The WebNLG example in Table 4 received the following scores, the total number of constituents being three: right: 2 (66%), wrong: 1 (33%), missed: 0 (0%), added: 1 (33%). If semantic information was repeated, it was rated as added.

The human evaluation analysis presented above is modest due to the lack of resources. To justify it, we argue that our focus is solely on semantic adequacy which is a more objective parameter in evaluations than, say, fluency or grammaticality. Furthermore, human scores showed strong correlations with most of automatic metrics. For example, *right* exhibits statistically significant correlations of 0.9, 0.55, 0.89, 0.85, -0.87 with BLEU, NIST, METEOR, ROUGE$_L$, SER respectively (Spearman's ρ; $p < 0.05$). *Wrong* has -0.91, -0.71, -0.88, -0.96, 0.78 correlation coefficients respectively.

With no intent to question the documented unreliability of automatic metrics in NLG, we attribute such high correlations to the design of our configurations which cover some extreme cases where models are supposed to show a drastic drop in performance.

3 Results and Discussion

We compared the output of the sequence-to-sequence model with attention described in Section 2.5 on two datasets (WebNLG and E2E) and considering eight different configurations depending on how rare words are handled (without delexicalisation, with delexicalisation, with a copy-and-coverage mechanism and with both delexicalisation and a copy-and-coverage mechanism) and on how the train/dev/test partition is constructed (unconstrained *vs.* constrained).

As pointed out in Section 2.6, automatic scores are reported using the whole test sets whereas human evaluation is based on shared MR instances between the non-constrained and constrained test sets (40 instances for E2E and 44 for WebNLG).

The results are summarised in Table 5 (E2E) and 6 (WebNLG). Some example predictions are shown in Tables 7 and 8.

	Unconstrained				*Constrained*			
	∅	*C*	*D*	*D+C*	∅	*C*	*D*	*D+C*
BLEU	0.54	**0.61**	0.56	0.56	0.09	0.34	0.44	**0.48**
NIST	9.70	**10.90**	10.19	10.11	2.37	6.81	7.37	**8.09**
METEOR	0.37	**0.42**	0.39	0.39	0.10	0.29	0.33	**0.36**
ROUGE$_L$	0.64	**0.71**	0.67	0.68	0.26	0.54	0.61	**0.65**
SER	43.66%	34.76%	34.93%	**31.83%**	92.5%	66.91%	50.48%	**45.45%**
right	69.26%	83.33%	83.70%	**87.04%**	10%	41.11%*	70.00%	**76.67%**
wrong	9.63%	5.56%	9.26%	7.78%	49.26%	32.59%	17.78%	15.93%
missed	21.11%	11.11%	7.04%	5.19%	40.74%	26.30%	12.22%	7.41%
added	0.37%	0%	0%	0%	1.11%	1.11%	0%	0%

Table 6: WebNLG dataset (*D*: Delexicalisation, *D+C*: delexicalisation and copying, *C*: copy and coverage, ∅: Neither copy nor delexicalsiation). The upper half of the table presents automatic evaluation results; the lower half—human evaluation results. Best scores are in bold. * – word repetitions present in predictions.

Delexicalisation and Copying *vs.* Standard Encoding-Decoding A first observation is that, when neither delexicalisation nor copying is used, there is a strong drop in semantic adequacy. In the worst case, the SER increases by 25.4 for the E2E dataset (constrained setting, ∅ *vs.* D+C) and by 47.05 points (constrained setting, ∅ *vs.* D+C) in the WebNLG dataset. Similarly, semantic adequacy descreases by up to 23.66 points for the E2E dataset (constrained setting, ∅ *vs.* D) and 47.5 points for the WebNLG dataset constrained setting, ∅ *vs.* D+C).

A similar, though weaker, trend can be observed for the remaining automatic metrics (e.g.,Δ BLEU:-0.12 points for the E2E dataset, in the unconstrained setting).

Delexicalisation, Copying or Both. The results show two trends. First, combining copying and delexicalisation yields the best results across the board. Second, while in the unconstrained setting, there is not much difference in terms of results between copying and delexicalisation, in the constrained setting, copying yields lower results (BLEU E2E:-0.15, BLEU WebNLG:-0.10, SA E2E:-13%, SA WebNLG:-29%, SER E2E:+8.91, SER WebNLG:+32.15). This suggests that copying only partially captures rare items. Looking at the outputs, it seems that copying seems to work better when the item to be copied has been seen in the training data. When an entity was not seen, the network often chooses to generate a frequent entity seen in the source, rather than copying. For instance, for the E2E data, restaurant names were not copied over in the constrained setting. In most cases, the input restaurant name was replaced by a restaurant name that is frequent in the training data. For example, given the MR *name[Cocum], eatType[coffee shop], near[The Rice Boat]*, the text *Near The Rice Boat there is a coffee shop called Fitzbillies* was generated, where *Fitzbillies*, a frequently occurring restaurant name in the training data (2,371 instances), wassubstituted for the input restaurant name *Cocum*.

Constrained vs. Unconstrained Setting There is a clear difference in terms of relative performance in the constrained *vs.* the unconstrained setting between the two datasets.

For the E2E dataset, since the constrained dataset is defined with respect to slot values (name and near) which are delexicalised, the constrained setting is in fact similar to the unconstrained setting. And indeed the scores are similar (e.g., BLEU:-0.05, SER:+1.10 and SA:-1% for the constrained setting). When using copying however, the results are lower in the constrained setting again suggesting that copying underperforms for items that have rarely been seen at training and development time (e.g., BLEU:-0.11, SER:+3.53 and SA:-13% for the unconstrained setting).

For the WebNLG data, the difference between constrained and unconstrained setting is much stronger for both delexicalisation and copying. For instance, for copying the BLEU score in the unconstrained setting is 0.61 *vs.* 0.34 in the constrained setting. Semantic adequacy also drops noticeably (unconstrained: 83%, constrained: 41%).

This is in line with Aharoni and Goldberg (2018)'s observation that in the unconstrained setting, the model learns to memorise association between facts and entities and thereby fails to generate text that adequately captures the meaning of the input data. The low scores for the copying mechanism also confirm the observation made above that copying underperforms for rare data fragments.

This Difference between datasets is further discussed in the next paragraph.

Semantic Adequacy. As mentioned above, the manual and automatic evaluation metrics we used to assess semantic adequacy strongly correlate. They both show that semantic adequacy is much lower for the WebNLG data (higher SER, higher proportion of wrong and missed items). This is not surprising since the WebNLG dataset contains a much higher number of distinct values (3648 against 79 in the E2E dataset) and exhibits a greater mismatch between input and output value names[8]. That is, the delta shows that the efficiency of copying and delexicalisation varies depending on the variety and content of the dataset.

The two datasets also differ with respect to the proportion of added slots which is higher for the E2E dataset and suggests an overfitting effect due to a skewed distribution in favor of inputs containing more than 3 attributes. Thus, the human evaluation shows that the majority of cases with added slots are cases where the input consists of three slots (the minimal number of attributes in E2E). The overgeneration can be explained by the restricted number of three-slot inputs in the E2E dataset (only 2.5% MRs out of the whole corpus). That claim is also supported by predictions produced by adversarial examples. While inputting dialogue moves consisting of 2 slots (the non-existent number of attributes in E2E), all eight E2E models tend to overgenerate by predicting texts with 3 or 4 slot-value pairs.

Fluency As mentioned in Section 2.6, while annotating the data for semantic adequacy, we found that almost all systems outputs were well-formed English sentences. The only exception was the WebNLG model with copy mechanism where stutterings were spotted in half of the examined instances. Despite those repetitions, it was always possible to detect the subject-predicate-object structure (e.g., *1001 kelvins is an escape velocity of 1001 kelvins*; *Asterix was created by R. Goscinny and was created by R. Goscinny*), so the annotation was not hampered.

[8]In the E2E dataset, the value name in the input usually is realised by the same string in the corresponding text while in WebNLG, they often differ e.g., USA/the United States of America).

4 Related Work

Delexicalisation remains one of the most popular techniques for handling rare named entities. We analysed the submissions participating in the E2E and WebNLG challenges, which used a neural approach. Among them, six teams applied delexicalisation (Chen et al., 2018; Davoodi et al., 2018; Juraska et al., 2018; Puzikov and Gurevych, 2018; Trisedya et al., 2018; van der Lee et al., 2017), three resorted to the copy mechanism (Chen, 2018; Elder et al., 2018; Gehrmann et al., 2018), two developed character-based systems (Agarwal and Dymetman, 2017; Deriu and Cieliebak, 2018), and another two made use of byte pair encodings (Elder, 2017; Zhang et al., 2018).

A copy mechanism allows to detect a word in an input sequence and to copy it to an output sequence. The copy mechanism is widely used in text production approaches where it is relevant for handling rare input but also, for instance in text summarisation, for copying input into the output. See et al. (2017), Gu et al. (2016), Cheng and Lapata (2016) introduced pointer networks (Vinyals et al., 2015) extended with a copy mechanism for text summarisation. Similarly, Cao et al. (2017) uses a copy mechanism to generate paraphrases and He et al. (2017) to generate answers. The copy mechanism is often paired with coverage, which tries overcoming a common problem of repeated or omitted content in neural network outputs. It was used for instance in NMT (Tu et al., 2016) and summarisation (See et al., 2017).

Finally, some approaches apply neither copying nor delexicalisation. In particular, Nayak et al. (2017), working in the restaurant domain for dialogue systems, investigated ways of including slot values directly into the input representation of sequence-to-sequence models.

5 Conclusion

We investigated the impact of copying and delexicalisation on two datasets and using two different ways of splitting the data into train, dev and test. The results show some regularities and highlight some interesting differences.

MR	name[Wildwood], eatType[pub], food[Italian], area[city centre], familyFriendly[no], near[Raja Indian Cuisine]
reference	Wildwood is an Italian pub situated in the city centre, near the Raja Indian Cuisine. It is not family-friendly.
unconstr, delex+	Wildwood is an Italian pub located in the city centre near Raja Indian Cuisine. It is not family-friendly.
unconstr, delex+, cc	Wildwood is an Italian pub located near the Raja Indian Cuisine in the city centre. It is not family-friendly.
unconstr, delex-	Wildwood is an Italian pub located in the city centre near **the Ranch**. It is not family-friendly.
unconstr, delex-, cc	Wildwood is an Italian pub located in the city centre near Raja Indian Cuisine. It is not family-friendly.
constr, delex+	Wildwood is an Italian pub near Raja Indian Cuisine in the city centre. It is not family-friendly.
constr, delex+, cc	Wildwood is an Italian pub located near the Raja Indian Cuisine in the city centre. It is not family-friendly.
constr, delex-	In the city centre near Raja Indian Cuisine there is an Italian pub called the **Aromi**. It is not family-friendly.
constr, delex-, cc	There is a pub near Raja Indian Cuisine in the city centre called **Aromi**. It serves Italian food and is not family-friendly.

Table 7: Example predictions for E2E. Mistakes are in bold.

MR	108 St Georges Terrace – floorCount – 50; 108 St Georges Terrace – completionDate – 1988
reference	108 St Georges Terrace has a floor count of 50 and was completed in 1988.
unconstr, delex+	108 St Georges Terrace has 50 floors and was completed in 1988.
unconstr, delex+, cc	108 St Georges Terrace has 50 floors and was completed in 1988.
unconstr, delex-	108 St Georges Terrace **cost 120 million Australian dollars**.
unconstr, delex-, cc	108 St Georges Terrace was completed in 1988 and has 50 floors.
constr, delex+	108 St Georges Terrace has 50 floors and was completed in 1988.
constr, delex+, cc	108 St Georges Terrace has 50 floors and was completed in 1988.
constr, delex-	**The coach of the Democratic Party in the United States is the Conservative Party (UK)**.
constr, delex-, cc	108 Georges Terrace **completionDate were created by 108 Georges**.

Table 8: Example predictions for WebNLG. Mistakes are in bold.

Overall, the results indicate that delexicalisation outperforms copying. Furthermore, they show that copying underperforms on rare items. Since delexicalisation is a somewhat ad hoc process, an interesting direction for future research would be to devise copying methods that are more accurate and that can better handle rare data items.

Another direction for future research would be to further investigate how the content and train/dev/test split of a dataset impact learning. Our results suggest two ways in which these may induce overfitting.

In the WebNLG dataset, strong associations between entities and facts seem to result in generation models that memorise facts with entities rather than generate a text that adequately verbalises the input. This is highlighted in the manual evaluation by the high number of wrong and missed data items observed both in the constrained and in the unconstrained setting.

In the E2E dataset, on the other hand, we saw that added facts are frequent and manual evaluation suggests that this is due to an overfitting effect whereby, because most inputs consists of more than three slot-value pairs, the models tend to overgenerate by predicting texts that verbalise four or more slot-value pairs.

In both cases, the copy-and-coverage mechanism does not suffice to ensure correct output and the results further decrease in the constrained setting. It would therefore be interesting to see to what extent better methods can be devised both for creating datasets and for devising train/dev/test splits that adequately test the ability of models to generalise.

Another direction for future work is to investigate the capability of byte pair encoding models and subword representations to handle rare input tokens in data-to-text generation.

References

Shubham Agarwal and Marc Dymetman. 2017. A surprisingly effective out-of-the-box char2char model on the e2e nlg challenge dataset. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 158–163. Association for Computational Linguistics.

Roee Aharoni and Yoav Goldberg. 2018. Split and rephrase: Better evaluation and stronger baselines. In *Proceedings of the 56th Annual Meeting of the*

Association for Computational Linguistics (Volume 2: Short Papers), pages 719–724. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2017. Joint copying and restricted generation for paraphrase. In *AAAI*, pages 3152–3158.

Mingje Chen, Gerasimos Lampouras, and Andreas Vlachos. 2018. Sheffield at e2e: structured prediction approaches to end-to-end language generation. Technical report, E2E Challenge System Descriptions.

Shuang Chen. 2018. A general model for neural text generation from structured data. Technical report, E2E Challenge System Descriptions.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494. Association for Computational Linguistics.

Elnaz Davoodi, Charese Smiley, Dezhao Song, and Frank Schilder. 2018. The e2e nlg challenge: Training a sequence-to-sequence approach for meaning representation to natural language sentences. Technical report, E2E Challenge System Descriptions.

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. Association for Computational Linguistics.

Jan Deriu and Mark Cieliebak. 2018. End-to-end trainable system for enhancing diversity in natural language generation. Technical report, E2E Challenge System Descriptions.

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*, HLT '02, pages 138–145, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Ondřej Dušek and Filip Jurcicek. 2015. Training a natural language generator from unaligned data. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 451–461. Association for Computational Linguistics.

Henry Elder. 2017. Adapt centre submission for the webnlg challenge. Technical report, WebNLG Challenge System Descriptions.

Henry Elder, Sebastian Gehrmann, Alexander OConnor, and Qun Liu. 2018. E2e nlg challenge submission: Towards controllable generation of diverse natural language. Technical report, E2E Challenge System Descriptions.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017a. Creating training corpora for nlg micro-planners. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 179–188. Association for Computational Linguistics.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017b. The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133. Association for Computational Linguistics.

Sebastian Gehrmann, Falcon Dai, Henry Elder, and Alexander Rush. 2018. End-to-end content and plan selection for natural language generation. Technical report, E2E Challenge System Descriptions.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640. Association for Computational Linguistics.

Shizhu He, Cao Liu, Kang Liu, and Jun Zhao. 2017. Generating natural answers by incorporating copying and retrieving mechanisms in sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 199–208. Association for Computational Linguistics.

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)*, pages 152–162. Association for Computational Linguistics.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72. Association for Computational Linguistics.

Chris van der Lee, Thiago Castro Ferreira, Emiel Krahmer, and Sander Wubben. 2017. Tilburg university models for the webnlg challenge. Technical report, WebNLG Challenge System Descriptions.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421. Association for Computational Linguistics.

Neha Nayak, Dilek Hakkani-Tr, Marilyn Walker, and Larry Heck. 2017. To plan or not to plan? discourse planning in slot-value informed sequence to sequence models for language generation. In *Proc. Interspeech 2017*, pages 3339–3343.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206. 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 of the Association for Computational Linguistics*.

Yevgeniy Puzikov and Iryna Gurevych. 2018. E2e nlg challenge: Neural models vs. templates. Technical report, E2E Challenge System Descriptions.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083. Association for Computational Linguistics.

Bayu Distiawan Trisedya, Jianzhong Qi, Rui Zhang, and Wei Wang. 2018. Gtr-lstm: A triple encoder for sentence generation from rdf data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1627–1637. Association for Computational Linguistics.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85. Association for Computational Linguistics.

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

Tsung-Hsien Wen, Milica Gasic, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721. Association for Computational Linguistics.

Biao Zhang, Jing Yang, Qian Lin, and Jinsong Su. 2018. Attention regularized sequence-to-sequence learning for e2e nlg challenge. Technical report, E2E Challenge System Descriptions.