Domain Adaptation for Natural Language Generation

Thesis Proposal

Zdeněk Kasner
Charles University, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Prague, Czech Republic
kasner@ufal.mff.cuni.cz

Abstract

The performance of natural language generation (NLG) systems varies across domains. This work aims to develop NLG systems that can perform well in the domains with the lack of available training data. We base our work on recent advances in transfer learning, using large pretrained neural language models to facilitate domain adaptation of NLG systems. In the thesis proposal, we first introduce the landscape of current approaches to NLG and the background for our work. Next, we report the results of our experiments, focusing on two challenges: (1) data-to-text generation with limited in-domain training data and (2) evaluating semantic accuracy of generated texts. Regarding future work, we outline our plans for improving the performance of neural NLG systems in more advanced scenarios by applying symbolic operations on an intermediate text plan.

1 Introduction

The information stored in structured data is commonly formulated in natural language to make the data easier for people to digest. Writing news articles or sports reports, generating weather forecasts, summarizing business statistics, or interpreting medical data—in all these tasks, structured data is transformed into text. Since this process generally requires language proficiency and in-domain knowledge, most texts are created manually by domain experts. However, given the present need for real-time user interactions with ever-growing data, this approach is not scalable enough.

The need can be solved by automatic approaches, which—if available—could take care of the routine tasks and provide valuable insights while being efficient both in terms of time and resources. Automatic text generation, better known as *natural language generation* (NLG), is one of the key tasks in natural language processing (NLP). Originally, NLG was used as a term for generating texts from structured data (Reiter and Dale, 1997). More opportunities for generating language have emerged over the years, including text summarization, question answering, or image captioning (Gatt and Krahmer, 2018), recently also accompanied by free-form text generation of stories and narratives (Radford et al., 2019; Rosa et al., 2021). Although these tasks may also be considered as a part of NLG, we will limit ourselves to the original interpretation NLG = *generating text from structured data*, now also referred to as data-to-text (D2T) generation.

Until the rise of neural models, a typical NLG system had to rely on several modules connected in a pipeline. As described in Section 2.1, these systems are still used nowadays in the majority of applications despite their narrow focus and high development costs since they can guarantee accurate outputs.

Neural models (Section 2.2) offer a new, data-centric alternative to the traditional approach. Models pretrained on large amounts of data can quickly adapt to various domains and generate texts with fluency comparable to human-written texts. However, other issues regarding semantic accuracy and controllability—a decisive factor in NLG—still constrain the models to experimental settings and prevents their practical deployment (Dale, 2020).

We aim to improve the performance of NLG systems in domains in which the current systems either perform poorly or require a considerable amount of human expertise. Towards that end, we address the following research questions:

(i) How to generate texts which are **fluent** across domains (i.e., grammatically correct and capturing domain-specific sentence style) having only a few in-domain training examples.

(ii) How to ensure that the text generated by neu-

ral models is **semantically accurate** (i.e., the semantics of the text corresponds to the input data).

(iii) How **evaluate** the performance of NLG systems in terms of semantic accuracy.

(iv) How to **unify the input data format** across domains to facilitate both the generation and the evaluation process.

(v) How to perform **symbolic operations** in NLG to allow more complex applications such as logical reasoning and integrating external knowledge.

(vi) How to **link the text and the data** to allow targeted improvements and ensure better interpretability of the generated text.

We have already conducted experiments regarding the research questions (i)-(iii) and we plan to address the questions (iv)-(vi) in our future work.

The thesis proposal is structured as follows: in Section 2, we start by summarizing the historical development of NLG regarding two competing paradigms: pipeline-based and end-to-end approaches. Next, we lay out the theoretical background for our work in Section 3, including an overview of the relevant models and datasets. In Section 4, we present our work regarding the fluency and semantic accuracy in NLG. In Section 5, we follow up with our experiments on the evaluation of generated texts. Regarding future work, we outline our plans on unifying the input data format, adding symbolic operations, and aligning the text with the data in Section 6. Finally, we summarize our work in Section 7.

2 Approaches to Natural Language Generation

2.1 Pipeline-based Approaches

A dominant approach to automatic text generation, which has prevailed since the beginning of the field, is to use several modules connected in a pipeline (Reiter and Dale, 1997, 2000; Gatt and Krahmer, 2018). The modules typically take care of the following tasks:

(1) *content determination* – deciding which facts to include in the text,

(2) *text structuring* – determining the order of the facts,

(3) *sentence aggregation* – dividing the facts into individual sentences,

(4) *lexicalisation* – transforming the facts to words and phrases,

(5) *surface realisation* – combining the words and phrases into a well-formed text.

An advantage of the pipeline-based approaches is that the modules are reusable and individual modules may be developed using different frameworks (template-based, grammar-based, statistical, etc.). The outputs of the systems are also explainable, which is a challenge for end-to-end approaches (Reiter, 2019). Building custom pipeline-based NLG systems is facilitated by frameworks such as SimpleNLG (Gatt and Reiter, 2009) or Data2Text Studio (Dou et al., 2018).

However, the modular architecture is accompanied by high development costs: creating the rules or templates requires considerable human effort, and the resulting system works only in the particular domain. The modular architecture also suffers from lower fluency—in part because of the rigidness of the specific approaches, in part because the errors accumulate along the pipeline (Castro Ferreira et al., 2019).

Practical applications of pipeline-based NLG systems range from generating information about train timetables (Aust et al., 1995) and weather forecasts (Goldberg et al., 1994; Reiter et al., 2005), to reporting about patients in health-care (Buchanan et al., 1995; Portet et al., 2009) or robo-journalism, i.e., generating news or sport stories (Chen and Mooney, 2008; Molina et al., 2011; Teixeira et al., 2020). The pipeline-based NLG systems also form the backbone of current commercial applications, including virtual assistants Amazon Alexa,[1] Apple Siri[2] or Google Home,[3] or applications for generating business intelligence reports.[4,5]

2.2 End-to-End Approaches

Recent advances in machine learning and neural networks—namely the encoder-decoder architecture (Sutskever et al., 2014), the attention mechanism (Bahdanau et al., 2015; Luong et al., 2015) and the Transformer model (Vaswani et al., 2017)—have brought the possibility to formulate NLG as a sequence-to-sequence (seq2seq) problem. This problem can be approached end-to-end, offering a counterpart to the modular approach. Moreover, the input-output transformations are learned from input-output pairs (with no need for explicit align-

[1] https://developer.amazon.com/alexa
[2] https://www.apple.com/siri/
[3] https://madeby.google.com/home
[4] https://www.arria.com/
[5] https://automatedinsights.com

ment), which reduces the amount of human effort and increases the robustness of the systems.

First neural approaches for NLG were based on **recurrent neural networks** (RNNs; Rumelhart et al., 1986). Wen et al. (2015) used a long short-term memory network (LSTM; Hochreiter and Schmidhuber, 1997; an extended variant of RNN) for generating responses in dialogue systems. Mei et al. (2016) added a content selection mechanism, which is trained jointly with the model. Dušek and Jurčíček (2016) showed that generating the text end-to-end is more efficient than a two-stage approach that uses an external surface realizer for deep syntax trees. Most recently, Gehrmann et al. (2018) added a pointer generator network (See et al., 2017), allowing the model to explicitly copy the parts of the input.

Various shared tasks and comparisons (Gardent et al., 2017b; Dušek et al., 2020; Castro Ferreira et al., 2019) showed that RNN-based approaches are generally competitive with pipeline-based approaches. The approaches differ in their strengths and weaknesses: the RNNs produce more fluent text, while the pipeline-based approaches can convey the data more accurately.

In 2017, models using the **Transformer architecture** (Vaswani et al., 2017) immediately became the state-of-the-art for high-resource NLP areas such as machine translation (Bojar et al., 2018) or constituency parsing (Kitaev and Klein, 2018). However, NLG as a low-resource area has only started to benefit from the Transformer architecture with the arrival of pretrained models, which allow efficient transfer learning on top of representations learned from large-scale unlabeled corpora (see Section 3.2). For example, the recent WebNLG+ shared task (Ferreira et al., 2020), in which the goal is to generate text from graph-structured triples (see Section 3.3), was dominated by systems based on pretrained denoising autoencoders (Yang et al., 2020; Agarwal et al., 2020; Kasner and Dušek, 2020b). The results suggest that Transformer-based pretrained models can produce outputs with considerably better fluency than previous NLG models. A subject of ongoing research (including Harkous et al., 2020; Len et al., 2020; Rebuffel et al., 2021; and our work) is to make the models perform well also in domains with a limited amount of training data and provide guarantees on the semantic accuracy of the output, so that the models can be deployed as real-world NLG systems.

3 Background

Here we first provide the theoretical background for the end-to-end approaches (Section 3.1), followed by an overview of relevant pretrained models (Section 3.2) and datasets (Section 3.3).

3.1 Architectures and Mechanisms

Language Models Given a sequence of tokens $\mathbf{y} = \{y_1, \ldots, y_n\}$, a language model (LM; Manning and Schutze, 1999) aims to output a probability of the sequence $p(\mathbf{y})$ in a language represented by the training corpus C. To learn the probability distribution p using a neural LM, we minimize the cross-entropy between the probability distribution of the model p_θ and the empirical distribution p of sequences in C:

$$H(p, p_\theta) = \mathbb{E}_p(-\log p_\theta) = -\sum_{\mathbf{y} \in C} p(\mathbf{y}) \log p_\theta(\mathbf{y}).$$

In practice, we factorize the probability distribution using the chain rule, conditioning the probability of a token on its left context:

$$p_\theta(\mathbf{y}) = \prod_{i=1}^{n} p_\theta(y_i | \mathbf{y}_{<i}).$$

After training, we can use the LM to generate sequences from left to right, e.g., by using greedy decoding, where we select the most probable token from the vocabulary V at each step:

$$y_i = \arg\max_{y \in V} p_\theta(y | \mathbf{y}_{<i}).$$

Masked Language Models In contrast to language modeling, masked language modeling (MLM; Devlin et al., 2019) allows the model to see the bidirectional context for each token. Randomly chosen tokens in the sequence $\mathbf{y}$ are replaced by a <*mask*> token and the model is trained to predict the original value of the masked tokens:

$$p_\theta(y_i | \mathbf{y}_{<i}, \text{<}mask\text{>}, \mathbf{y}_{>i}).$$

Since MLMs need both left and right context, the models cannot be straightforwardly used for sequence generation. However, the bidirectional context allows to learn contextual representations for tokens, which can be used by subsequent layers, e.g. for token classification tasks.

Encoder-Decoder In the encoder-decoder architecture, an input sequence $\mathbf{x} = \{x_1, \ldots, x_n\}$ is processed by the encoder into a sequence of hidden states $\mathbf{h} = \{h_1, \ldots, h_n\}$. The decoder then uses $\mathbf{h}$ to decode the sequence of output tokens $\mathbf{y} = \{y_1, \ldots, y_m\}$. Both the encoder and the decoder typically consist of mutliple layers of feed-forward neural networks. We can use the encoder-decoder architecture for seq2seq generation by computing the conditional probability of $\mathbf{y}$ given $\mathbf{x}$, conditioning on the left context:

$$p_\theta(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} p_\theta(y_i|\mathbf{y}_{<i}, \mathbf{x}).$$

In this light, a language model can be seen as decoder-only, whereas a masked language model as encoder-only.

Attention Mechanism An attention mechanism (Bahdanau et al., 2015; Luong et al., 2015) enables a model to incorporate relevant information from a previous sequence of hidden states $\mathbf{h}'$ into the value of the state h_i. The output of the attention mechanism is a context vector c_i, which is a weighted combination of $\mathbf{h}'$:

$$c_i = \sum_{j=1}^{m} \alpha_{ij} h'_j.$$

In RNNs, c_i is typically summed with a previous hidden state h_{i-1} before being passed through the non-linear function of the network.

Transformer Architecture The Transformer (Vaswani et al., 2017) is a multi-layer encoder-decoder model. The Transformer generalizes the attention mechanism using three vectors: queries Q, keys K, and values V:

$$att(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V.$$

The dot product between Q and K computes the relatedness score of the pair of states, while V is used for computing the actual attention value (d_k is the dimension of the keys used for normalization). Q, K and V may also come from the same layer, which means the model attends to the layer itself. This concept, called *self-attention*, allows the Transformer to parallelize the computation of states in each layer while preserving the dependencies between tokens.

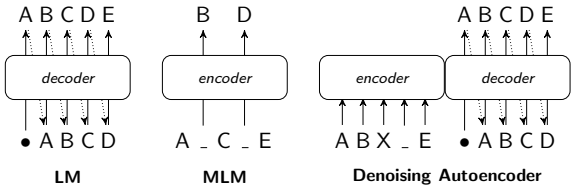

Figure 1: A scheme of the common objectives used by pretrained models: 1) language modeling (decoder-only), 2) masked language modeling (encoder-only), 3) denoising (encoder-decoder). The special symbol • (beginning of a sentence) is used to bootstrap the decoding process.

Denoising Autoencoders An *autoencoder* (Ballard, 1987) deals with a specific instance of the seq2seq problem in which the input and output sequence is identical, i.e. $\mathbf{y} = \mathbf{x}$. The aim of the model, based on the encoder-decoder architecture, is to learn a compact and informative representation $\mathbf{h}$ which allows reconstructing the input. A *denoising autoencoder* (Vincent et al., 2008) is an autoencoder variant which takes as an input a corrupted version of the original sequence $\tilde{\mathbf{x}}$ and aims to restore the original undistorted input $\mathbf{x}$. Besides adding the capability to remove noise from the input, this approach increases the robustness of $\mathbf{h}$ to input perturbations.

Self-Supervised Learning The models for the (masked) language modeling and denoising are trained using *self-supervised learning* paradigm (Schmidhuber, 1990). As the name suggests, the training labels are derived automatically from the unlabeled training data. In the case of self-supervised language modeling, the labels are equal to the original tokens (e.g., the model is given a <*mask*> token, while it is trained to predict the original token). This paradigm allows efficient training of large models from unlabeled corpora. Figure 1 shows a comparison of the self-supervised objectives.

3.2 Pretrained Models

In our context, the term *pretrained models* refers to the family of models using the Transformer architecture which were trained on large corpora using self-supervised learning. Pretraining is the key to **transfer learning**, i.e., re-using the parameters of the model for a downstream task. The parameters usually contain useful language representations and need to be only slightly adjusted to achieve good performance on downstream tasks. The process of

adjustment, called *finetuning*, involves additional training of the model on task-specific examples. In some cases, it may be beneficial to freeze a subset of the parameters, but usually, it is possible to finetune all the parameters at once (Peters et al., 2019; Rothe et al., 2020).

The following models are used in our work and belong among the most influential (see, e.g., Qiu et al., 2020 for further reference). All of the presented models are freely accessible in the Huggingface Transformers repository (Wolf et al., 2019).[6]

BERT BERT (Bidirectional Encoder Representations from Transformers; Devlin et al., 2019) was the first pretrained language model based on the Transformer architecture. It builds upon previous work in contextualized representations (Peters et al., 2018) and transfer learning (Howard and Ruder, 2018). BERT uses the Transformer encoder to output a contextualized representation of each token. A representation for a special *[CLS]* token at the beginning of the sequence is used for sequence classification. The model is trained on BooksCorpus (Zhu et al., 2015) and English Wikipedia using the MLM objective, masking or replacing 15% of the input tokens. Using BERT as a backbone model has brought state-of-the-art results in various NLP areas (e.g., Devlin et al., 2019; Liu and Lapata, 2019; Joshi et al., 2020). Although follow-up work has revealed inefficiencies in training the model (Liu et al., 2019; Lan et al., 2019), BERT is often used as a prototypical pretrained language model for further investigations (e.g., Petroni et al., 2019; Rogers et al., 2020; Limisiewicz et al., 2020; Izsak et al., 2021).

RoBERTa RoBERTa (Liu et al., 2019) shares the architecture with BERT, but achieves better performance thanks to carefully selected hyperparameters and increased training data size. At the time of writing, finetuned versions of the model achieve state-of-the-art results in NLP tasks such as natural language inference, grammatical error correction, and common sense reasoning (Liu et al., 2019; Omelianchuk et al., 2020).

GPT-2 GPT-2 (Radford et al., 2019) is a standalone Transformer decoder which is capable of conditional left-to-right text generation. The model allows calculating the perplexity of a sentence, which we use in our work as an indirect measure of its fluency (see Section 4.1). Unlike the more recent and much larger model GPT-3 (Brown et al., 2020), GPT-2 is openly accessible.

BART BART (Lewis et al., 2020a) is a pretrained denoising autoencoder. The model is trained to denoise the input sequence corrupted by various transformations, including token masking, token or span deletion, and sentence permutation. Unlike the aforementioned models, BART uses the full Transformer encoder-decoder architecture, which makes it suitable for seq2seq generation, including NLG, text summarization, and question answering.

T5 The T5 model (Raffel et al., 2020) shares many characteristics with BART, including the encoder-decoder architecture, denoising objective, and comparable performance on seq2seq tasks. Its specific feature is a unified text-to-text format for each task utilizing prompts (e.g., *"summarize: "* or *"translate English to French: "*), which allows to pretrain the model on multiple downstream tasks at once.

Multilingual Models The original versions of the aforementioned models are English-only, as data in English is easily accessible online and it is the dominant language for NLP benchmarks (e.g., Wang et al., 2019; Gehrmann et al., 2021). Nevertheless, there are also versions of the models with identical architecture trained on multilingual corpora, such as XLM-RoBERTa (Conneau et al., 2020; 100 languages), mBART (Liu et al., 2020a; 25 languages), or mT5 (Xue et al., 2021; 101 languages). An advantage of the multilingual models is the possibility of cross-lingual transfer, i.e., training for a task in a high-resource language and applying the model to the same task in a low-resource language.

3.3 Datasets

Datasets are an important asset in NLG, providing two resources: a) training examples and b) evaluation benchmarks. Moreover, a dataset is usually focused on particular areas of interest, which gives us a notion of a **domain**.[7] Since structured data paired with text descriptions is generally not available on

[6] https://huggingface.co/transformers/

[7] In NLG, a *domain*, in the sense of an *area of knowledge or activity* (Merriam-Webster, 2021), is usually considered to be an application area (Budzianowski et al., 2018; Rastogi et al., 2020; van der Lee et al., 2020). A dataset may cover multiple domains, but the examples usually follow the same input format and sentence style. In our work, we aim towards systems that also generalize to domains across multiple datasets.

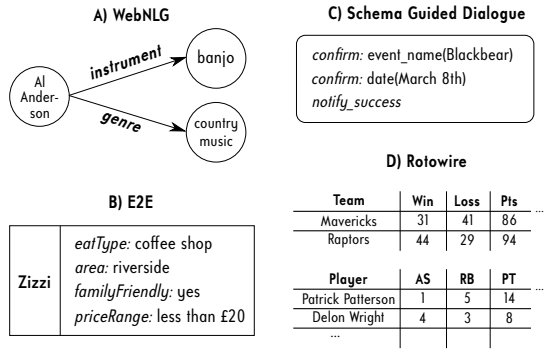

Figure 2: Input data formats in NLG datasets: a graph (A), a set of key-value pairs (B, C), and a set of tables (D). The target (not shown here) is a textual description which corresponds to the input data (A,B,C) or which describes selected important content (D).

the web (or not in sufficient quality), the datasets are created manually or semi-automatically with the help of crowdsourcing platforms such as Amazon Mechanical Turk.[8] The following datasets (illustrated in Figure 2) are commonly used as benchmarks in NLG research. We also use the datasets in our experiments described in Section 4 and 5.

WebNLG The WebNLG dataset (Gardent et al., 2017a,b) contains graph-structured RDF triples from DBPedia (Auer et al., 2007) and their crowdsourced descriptions. An RDF triple has three constituents: a *subject* and an *object* (usually describing entities such as people, objects, or places), and a *predicate*, expressing the relation between the subject and the object. The WebNLG Challenge (Gardent et al., 2017b; Ferreira et al., 2020) is a shared task on RDF-to-text generation, which evaluates the systems on the WebNLG dataset. We describe our submission to the 2020 round of the challenge in Section 4.2.

E2E The E2E dataset (Dušek et al., 2020) contains restaurant descriptions in the form of attribute-value pairs and corresponding human-written recommendations. The name of the dataset is derived from the E2E Challenge, a shared task that focused on evaluating end-to-end NLG systems. Dušek et al. (2019) show that the original version of the dataset contains semantic noise (incorrect or missing facts in the crowdsourced descriptions) and present a cleaned version of the dataset, which we use for our experiments.

[8] https://www.mturk.com

Schema Guided Dialogue Schema Guided Dialogue (SGD; Rastogi et al., 2020) is a dataset with task-oriented dialogues. Each dialogue consists of system and user utterances, together with system actions for each turn. Following Kale and Rastogi (2020a), we use the dataset as an NLG benchmark by generating system utterances from the system actions.

Rotowire Rotowire (Wiseman et al., 2017) is a dataset with tabular statistics of basketball games and their corresponding textual summaries, in which only a relevant subset of the input data should be verbalized. Together with the above-average length of the target summaries, this aspect makes the dataset particularly challenging for NLG systems. We evaluate the outputs of neural models for the Rotowire dataset in Section 5.2.

4 Domain Adaptation for NLG

In this section, we describe our experiments regarding domain adaptation for natural language generation:

(1) iterative text generation with text-editing models (Section 4.1),
(2) multilingual NLG using denoising autoencoders (Section 4.2),
(3) task-specific pretraining for low-resource NLG (Section 4.3).

We focus here on research questions (i) and (ii), i.e., generating a fluent text which also verbalizes all the required facts, particularly in the domains with few or zero training examples. In order to use neural LMs, we formulate the NLG as a seq2seq problem. In Section 4.1 and 4.3, we also utilize simple template-based transformations to take advantage of the text-to-text pretraining.

4.1 Iterative Text Editing

This section is based on our work published in Kasner and Dušek (2020a). Our idea is to transform individual data items to text using trivial templates (which are accurate but not fluent) and let a neural model improve the resulting text. With this approach, we prioritize semantic accuracy, but we still leverage the language capabilities of a pretrained LM.

The approach is illustrated in Figure 3. Let us consider input data X = {*(Dublin, capital, Ireland), (Ireland, language, English)*}. We can transform the first triple into text by filling the template <*subject*> *is the capital of* <*object*> . In the next step,

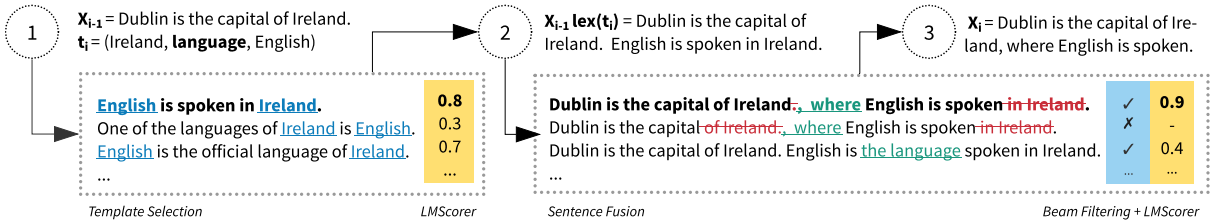

Figure 3: Our approach to data-to-text generation, focusing on semantic accuracy of the output. We transform the data to text using simple templates and iteratively improve the resulting text using a text-editing model.

we concatenate it with the second template <*subject*> *is spoken in* <*object*>. To improve the fluency of the text, we process it with a text-editing model trained to fuse sentences. We filter out the outputs from the model missing any facts, rescore the remaining outputs according to their fluency, and use the best result as the output. We repeat the process until we transform all the input triples.

Our text-editing model is based on LaserTagger (Malmi et al., 2019), which is a BERT-based encoder adapted for text editing. LaserTagger generates the output sequence by tagging the input tokens by the tags *KEEP*, *DELETE*, or *ADD* a phrase before the token. The main advantage of the text-editing approach is the reduced size of the vocabulary (containing mostly function words or phrases), which limits the possibility of generating false facts. We check for missing facts using simple heuristics (literal string matching or regular expressions). For rescoring the sentences, we compute a geometric mean of the token conditional probability using an off-the-shelf GPT-2 model. We derive the training data for our text-editing model automatically from the WebNLG and E2E datasets, using pairs of examples where one of the examples contains a single extra triple.

The fluency of our system lags behind state of the art in terms of automatic metrics, although our fusion component still improves the results compared to the baseline with no fusion. The strength of our system is in 100% coverage of input facts since a fallback to a simple template is used every time a fact is missing. Our system also allows fine-grained control over the generation process and shows how to formulate data-to-text generation via text-editing operations, which we aim to follow in our future work (cf. Section 6.3).

4.2 Multilingual NLG with Denoising Autoencoders

This section describes our submission for the WebNLG+ Challenge (Ferreira et al., 2020) published in Kasner and Dušek (2020b). We participated in the track focusing on generating text from the RDF triples in the WebNLG dataset. For the challenge, the dataset (originally English-only) was expanded by Russian reference texts, and the participants were encouraged to submit models for both languages.

Inspired by the good performance of pretrained denoising autoencoders on seq2seq tasks, we submitted a solution that used a simple and identical setup for both English and Russian. For each language, we finetuned a multilingual denoising autoencoder (mBART; Liu et al., 2020b) on a linearized[9] version of the WebNLG data. Our model placed in the first third of the leaderboard for English and first or second for Russian on automatic metrics, and in the best or second-best system cluster on human evaluation. The model performed well even on domains that were not part of the training set, although the performance was lower than for the seen domains.

The shared task has shown that pretrained models for seq2seq generation can excel on simpler NLG datasets (i.e., without content selection), beating more complex approaches that consider the data structure. However, our manual analysis has revealed details which can be still improved upon, such as correct verbalization of predicates in unseen domains (e.g., understanding that the predicate *populationMetro* means the number of inhabitants of the city), correct understanding of the directions of the relations (e.g., *follows* vs *isFollowedBy*), and occasional hallucinations of facts

[9] Markers <*s*>, <*p*>, and <*o*> were used for separating the constituents of each triple; the triples were concatenated in their default order.

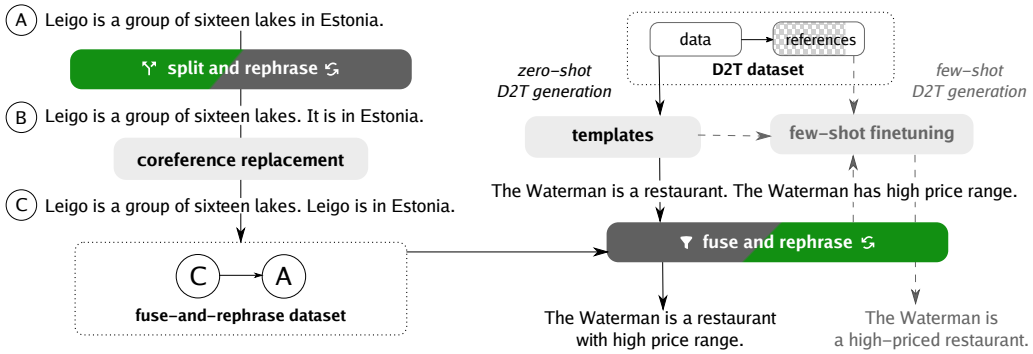

Figure 4: An overview of our fuse-and-rephrase approach. *Left*: we build a fuse-and-rephrase dataset from a large, unlabeled corpus and use it to train a fuse-and-rephrase model. *Right*: We first use a set of templates to transform the data to text. Then we apply the fuse-and-rephrase model on the templates direcly (full arrows) or we finetune the model with a few in-domain examples (dashed arrows).

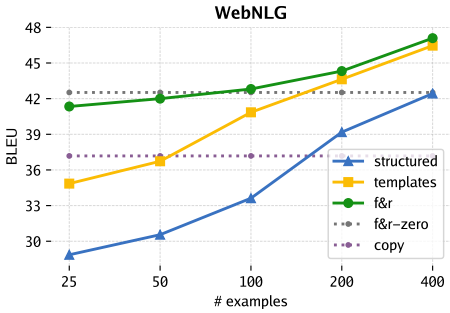

Figure 5: Results of our fuse-and-rephrase model (Section 4.3) on the WebNLG dataset in terms of BLEU score (Papineni et al., 2002). Our model (green/circles) is able to achieve better score with a small number of examples than the model based solely on templates (yellow/squares) and the model using structured input (blue/triangles).

not present in the input data. We also assume that the performance would be lower for languages that are less represented in the pretraining corpus than English and Russian.

4.3 Fuse and Rephrase

This section describes our unpublished work, which is currently under review for the 2021 Conference on Empirical Methods in Natural Language Processing.

Building upon Kale and Rastogi (2020b), Ribeiro et al. (2020) and our work described in Section 4.2, we note that denoising autoencoders are efficient in generating text using a simple linearized representation of the structured data. However, the models tend to overfit the particular representation, leading to inconsistent results when applied to new domains. Kale and Rastogi (2020a) have shown that these problems can be partially mitigated by first transforming data to text using trivial, single-attribute templates (similarly to our work in Section 4.1), allowing the model to work only with text.

In our work, we devise a task-specific pretraining task (Gururangan et al., 2020) on which the model can be trained for better performance in few-shot settings. The task, which we name **fuse-and-rephrase**, is inspired by the process of transforming templates to fluent text. It is roughly inverse to the split-and-rephrase task, which aims to split a complex sentence into multiple simpler sentences (Narayan et al., 2017; Botha et al., 2018). We show that the data for the task can be generated automatically from an unlabeled corpus (e.g. Wikipedia) by applying a split-and-rephrase model on the corpus and then swapping the data, i.e., using the split sentences as sources and the original sentences as targets (see the illustration of the process in Figure 4).

We evaluate the benefits of our task by generating texts using a T5 model (Raffel et al., 2020) in three setups: (1) the model which uses the templates and is pretrained for the fuse-and-rephrase task, (2) the model which uses only the templates, and (3) the model which uses a linearized structured representation (note that our task applies on text-based input only). Our results on three datasets (WebNLG, E2E, SGD; see also Figure 5) show that our additional pretraining task helps to improve output quality regarding both fluency and semantic accuracy in few-shot settings (i.e., with a limited number of in-domain examples), and the model can perform well even in zero-shot settings (i.e., with no in-domain examples). The models using templates perform better than the model using structured representation, which can be explained by the

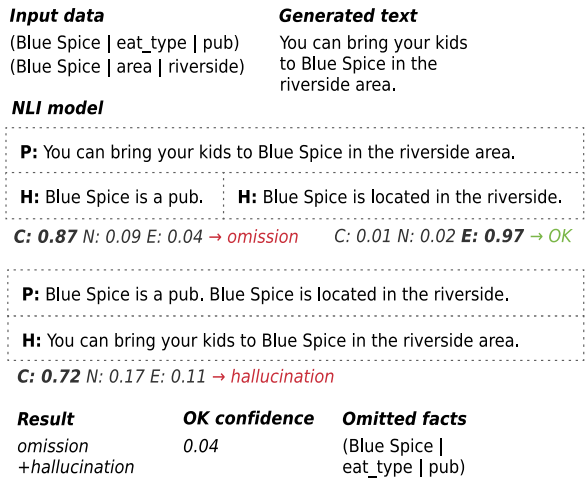

Figure 6: Our method for evaluating semantic accuracy of a generated sentence using RoBERTa for NLI (Section 5.1). To detect omissions, we check if the sentence entails individual facts; to detect hallucinations, we check if the concatenated facts entail the sentence.

fact that the templates provide more semantic information. However, the templates still have to be created manually, which we plan to improve upon. The model can also be trained to follow a plan of order and aggregation of the facts, which we plan to use for more fine-grained control over the output (see Section 6.1 for both).

5 Evalution of NLG

In this section, we describe our experiments on evaluating semantic accuracy of texts generated from data:

(1) on a sentence level using a model trained for natural language inference (Section 5.1),
(2) on a token level using a custom model (Section 5.2).

We focus on the research question (iii), i.e., developing automatic metrics for evaluating semantic accuracy in NLG. Currently, this aspect can be measured only indirectly by other metrics (which is imprecise), using a slot-error script (which is domain-specific), or using human evaluation protocols (which is time-consuming and costly). Automatic metrics will facilitate the development of NLG systems faithful to the input data, which is a subject of current research, including our work in Section 4.

5.1 Evaluating Semantic Accuracy using Natural Language Inference

This section is based upon our work published in Dušek and Kasner (2020). In the work, we propose an automatic method for detecting omissions (i.e., missing facts) and hallucinations (i.e., extra facts) in the text generated from data.

We build our system upon a model trained for natural language inference (NLI). A NLI model takes two inputs—a *hypothesis* and a *premise*—and produces one of the three outputs: the hypothesis is *entailed* by (follows from) the premise, *contradicts* the premise, or their relation is *neutral*. We propose that if the input data correspond to the generated text, individual facts must be entailed by the text (otherwise a fact is missing), and also the text should be entailed by the concatenation of individual facts (otherwise there is a hallucination in the text). Since NLI models are trained only on the text modality, the key to our approach is again to transform individual data items into text using simple templates, similarly to our work in Section 4.1 and 4.3. Our method is illustrated in Figure 6.

For our experiments, we use an off-the-shelf RoBERTa model (Liu et al., 2019) trained on the MNLI dataset (Williams et al., 2018) without any additional finetuning. Our results on the WebNLG and E2E datasets show that our metric can achieve high accuracy in identifying erroneous system outputs (77% and 91%, respectively) using human annotations or a slot-error script as a ground truth. Moreover, our manual analysis revealed that out of the examples where the output of our metric differed from the ground truth, around a half were classified correctly by our metric, and the error was in the ground truth data. The results suggest that our metric can be used for automatic evaluation of semantic accuracy of generated texts, which will allow more rapid development of NLG systems.[10]

5.2 Token-Level Error Checking

This section describes our system for the Shared Task in Evaluating Accuracy (Reiter and Thomson, 2020). The goal of the shared task is to detect semantic errors in texts generated by neural systems on the Rotowire dataset. Errors should be classified on token level using predefined categories such as *incorrect number*, *incorrect name*, *context error*,

[10] As an early example, the metric helped us to evaluate the semantic accuracy of our system in Section 4.3 (together with specialized slot-error scripts).

etc. A system for detecting the errors on token level will facilitate the development of reliable NLG systems and potentially allow to remove the incorrect facts from the generated output.

Our system is illustrated in Figure 7. We first generate text descriptions of the facts which can be derived from the input table using a rule-based NLG system (Mille et al., 2019). Since the set of generated sentences is much larger than the maximum input size of our error-checking model, we select a subset of relevant sentences by measuring cosine similarity over sentence embeddings from Sentence Transformers (Reimers et al., 2019). The sentences with the highest similarity score are concatenated with the evaluated sentence and provided as an input to the error checking model.

For error checking, we use a RoBERTa model with token classification head, which we finetune for the task. Our final submission is based on a two-stage approach. First, we train the model on custom "corrupted" data which were created by adding errors to the Rotowire training data (containing 3,395 examples), and then we finetune the model on the annotated data provided for the shared task (containing only 60 examples). According to preliminary results provided by the organizers, our system achieves 69% recall and 75% precision on the test set, which makes our system the best out of three submitted automatic metrics.

6 Research Plan

In this section, we describe show how we plan to address the research questions (iv), (v), and (vi) (see Section 1) in our future work. In Section 6.1, we will outline our initial plans on developing a unified representation using an explicit text plan which will serve as an input for a pretrained LM. In Section 6.2, we will discuss several potential extensions to the text plan, including logical reasoning and external knowledge retrieval. Finally, in Section 6.3 we describe our research ideas on alignment between the text and the data, which can be developed in parallel and eventually integrated with our previous research.

6.1 Unifying the Input Data Representation

The variations in input data format are one of the major obstacles to domain adaptation in NLG. Although a simple linearization of the input data for a seq2seq model has been shown to bring better results than specialized approaches such as graph neural networks (Zhao et al., 2020; Ribeiro et al., 2020; Kale and Rastogi, 2020b), the models require a large number of training examples and do not generalize to different input representations. In some cases, the input data may also contain either too much information, forcing the model to perform implicit content selection (Lebret et al., 2016; Wiseman et al., 2017), or not contain enough information, forcing the model to introduce extra knowledge in the text (van der Lee et al., 2020). Both cases lead to undesirable omissions or hallucinations.

We will base our initial experiments on a two-step approach: first transforming the input data to a unified text plan, which should contain *all* and *only* the information necessary for generating the text, and subsequently verbalizing the text plan with a neural LM. The text plan will be structured and will include additional annotations together with explicit control codes for the LM. As demonstrated in work on controllable generation, including control codes enables fine-grained control over the output of the model (Len et al., 2020; Keskar et al., 2019; Ficler and Goldberg, 2017). At the same time, the structure of the text plan will be linearizable, which will allow us to directly use the architectures from our previous work.

Our approach will build upon research on explicit content planning (Moryossef et al., 2019; Elder et al., 2019; Trisedya et al., 2020). Unlike these works, we plan to experiment with assembling a rich representation of the input compiled from multiple sources, including ordering and aggregation of the input facts, retrieved external knowledge (see Section 6.2), and explicit linguistic information (e.g., tense) estimated from the training data. In addition, we will aim to unify the input format, e.g., by transforming individual data items to text using templates, similarly to our work in Section 4 and 5. We will also experiment with generating the templates automatically using the approach of Laha et al. (2019).

6.2 Adding Symbolic Knowledge Processing

The use of low-level objective functions for training the neural LMs (cf. Section 3) limits the use of the LMs in more complex scenarios, including generating longer texts, such as full articles or stories (Fan et al., 2019; See et al., 2019; Rosa et al., 2021), and generating texts which require logical or common sense reasoning (Chen et al., 2020a,b;

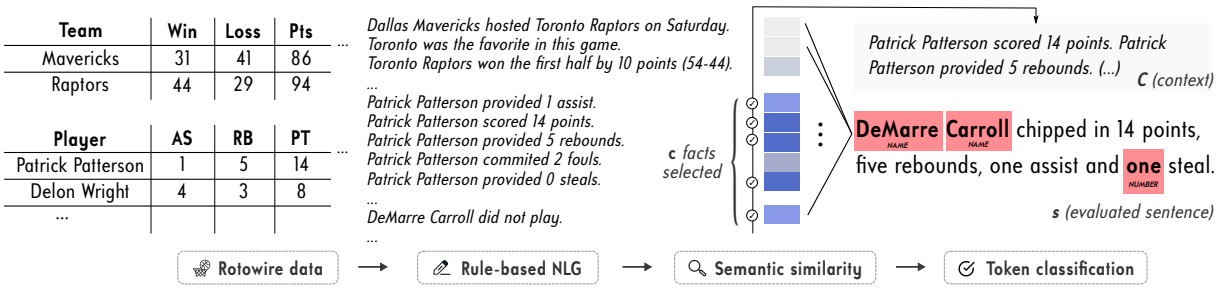

Figure 7: Our approach to detecting semantic errors on token level. First, we generate facts from the input table using a rule-based NLG system. For each evaluated sentence, we select relevant facts (according to the semantic similarity score) and use the facts as a context for a pretrained LM, which is trained to annotate the data.

Lin et al., 2020). Without additional grounding, the output in these scenarios—often based on spurious correlations in the training data—may be biased and incoherent (McCoy et al., 2019; Bender et al., 2021).

In contrast, the explicit nature of the text plan in Section 6.1 allows us to control the output of the model using symbolic operations over the text plan. We plan to conduct experiments in these research directions:

- Fetching information from **external knowledge bases** in order to include additional context about factual data (Bollacker et al., 2008) or commonsense knowledge (Speer et al., 2012; Hwang et al., 2020), e.g., retrieving additional context for named entities. The goal is to introduce relevant facts in the text without retraining the model and relying on its implicit knowledge, similarly to (Lewis et al., 2020b).
- Using symbolic reasoning operations such as **logical inference** and **numerical computations**. The operations may be carried on explicitly on the text plan. The goal is to limit the dependence on the logical and quantitative reasoning capabilities of neural LMs, which are not reliable (Andor et al., 2019; Geva et al., 2020).

Our general aim is to use dedicated modules for tasks that have inherently symbolic nature while leveraging good surface realization capabilities of neural models. In contrast to pipeline-based approaches, we will assemble individual pieces of information for the text plan independently and use the information as an input for the neural model, which will take care of the final generation step.

6.3 Linking the Text with the Data

The neural NLG in the current form is a one-sided transformation process—the individual parts of the output sequence cannot be reversely related to the input. Although there is research on interpreting the attention values of the models (Vashishth et al., 2019), research is lacking on applying the results further beyond explaining the reasoning of the model since clues in attention values are weak (Serrano and Smith, 2019; Thorne et al., 2019; Li et al., 2020).

We aim for a more **explicit alignment** between the data and the text, similarly to phrase-based machine translation systems (Och and Ney, 2003). We hypothesize that adding explicit connections will not only increase the interpretability of the models but also allow informed and targeted improvement of the text. This line of research is orthogonal to our other research directions, but it may benefit from the positive results and eventually be integrated into a unified system.

We want to experiment with multiple ways of extracting the alignments. The first option is to use unsupervised alignment approaches based on statistical (Och and Ney, 2003) or neural models (Garg et al., 2019; Zenkel et al., 2020). The approaches provide sufficient accuracy for the sentence-level alignment and also allow a more fine-grained, phrase or token-level alignment. Another option is to use the duality of NLG and natural language understanding (NLU)—an inverse process of parsing a text in natural language into structured representation (Su et al., 2020). In this case, we would use the explicit text plan outlined in Section 6.1. The generated text will be parsed to structured representation, which will be subsequently aligned with the text plan, e.g., using graph matching algorithms (Conte et al., 2004; Caetano et al., 2009).

Regarding the targeted improvement of the text,

we plan to follow up on our experiments from Section 4.1 using **text-editing models** which we will modify for our purposes. In contrast to full seq2seq models, text-editing models allow to control and interpret individual edits over multiple iterations. The most recent text-editing models (Mallinson et al., 2020; Stahlberg and Kumar, 2020) also allow reordering of individual tokens, which could help to alleviate the problems with fluency from our previous experiments.

7 Conclusion

The thesis proposal described current challenges in adapting NLG systems to new domains. We showed how pretrained neural LMs allow us to tackle some of these challenges while posing new research problems. In our experiments, we first addressed the problem of fluency and semantic accuracy of texts generated by neural NLG systems, focusing on low-resource domains. Semantic accuracy was also the subject of our research regarding NLG evaluation, in which we developed automatic metrics for evaluating the semantic accuracy of generated texts. In the future, we plan to follow up on our experiments by unifying the input to the neural models, combining advantages of symbolic operations and surface realization capabilities of pretrained LMs to improve the reliability and domain independence of NLG systems.

References

Oshin Agarwal, Mihir Kale, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2020. Machine translation aided bilingual data-to-text generation and semantic parsing. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 125–130.

Daniel Andor, Luheng He, Kenton Lee, and Emily Pitler. 2019. Giving bert a calculator: Finding operations and arguments with reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5947–5952.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Harald Aust, Martin Oerder, Frank Seide, and Volker Steinbiss. 1995. The philips automatic train timetable information system. *Speech Communication*, 17(3-4):249–262.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Dana H Ballard. 1987. Modular learning in neural networks. In *AAAI*, volume 647, pages 279–284.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 conference on machine translation (WMT18). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 272–303, Belgium, Brussels. Association for Computational Linguistics.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Jan A Botha, Manaal Faruqui, John Alex, Jason Baldridge, and Dipanjan Das. 2018. Learning to split and rephrase from wikipedia edit history. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 732–737.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Bruce G Buchanan, Johanna D Moore, Diana E Forsythe, Giuseppe Carenini, Stellan Ohlsson, and Gordon Banks. 1995. An intelligent interactive system for delivering individualized information to patients. *Artificial intelligence in medicine*, 7(2):117–154.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Tibério S Caetano, Julian J McAuley, Li Cheng, Quoc V Le, and Alex J Smola. 2009. Learning graph matching. *IEEE transactions on pattern analysis and machine intelligence*, 31(6):1048–1058.

Thiago Castro Ferreira, Chris van der Lee, Emiel van Miltenburg, and Emiel Krahmer. 2019. Neural data-to-text generation: A comparison between pipeline and end-to-end architectures. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 552–562, Hong Kong, China. Association for Computational Linguistics.

David L Chen and Raymond J Mooney. 2008. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*, pages 128–135.

Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. 2020a. Logical natural language generation from open-domain tables. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7929–7942, Online. Association for Computational Linguistics.

Zhiyu Chen, Wenhu Chen, Hanwen Zha, Xiyou Zhou, Yunkai Zhang, Sairam Sundaresan, and William Yang Wang. 2020b. Logic2Text: High-fidelity natural language generation from logical forms. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2096–2111, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Donatello Conte, Pasquale Foggia, Carlo Sansone, and Mario Vento. 2004. Thirty years of graph matching in pattern recognition. *International journal of pattern recognition and artificial intelligence*, 18(03):265–298.

Robert Dale. 2020. Natural language generation: The commercial state of the art in 2020. *Natural Language Engineering*, 26(4):481–487.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Longxu Dou, Guanghui Qin, Jinpeng Wang, Jin-Ge Yao, and Chin-Yew Lin. 2018. Data2Text studio: Automated text generation from structured data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 13–18, Brussels, Belgium. Association for Computational Linguistics.

Ondřej Dušek, David M Howcroft, and Verena Rieser. 2019. Semantic noise matters for neural natural language generation. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 421–426.

Ondřej Dušek and Filip Jurčíček. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 45–51.

Ondřej Dušek and Zdeněk Kasner. 2020. Evaluating semantic accuracy of data-to-text generation with natural language inference. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 131–137.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2020. Evaluating the state-of-the-art of end-to-end natural language generation: The E2E NLG challenge. *Computer Speech & Language*, 59:123–156.

Henry Elder, Jennifer Foster, James Barry, and Alexander O'Connor. 2019. Designing a symbolic intermediate representation for neural surface realization. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 65–73, Minneapolis, Minnesota. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2650–2660, Florence, Italy. Association for Computational Linguistics.

Thiago Ferreira, Claire Gardent, Nikolai Ilinykh, Chris van der Lee, Simon Mille, Diego Moussallem, and Anastasia Shimorina. 2020. The 2020 bilingual, bi-directional WebNLG+ shared task overview and evaluation results (WebNLG+ 2020). In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. In *Proceedings of the Workshop on Stylistic Variation*, pages 94–104.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017a. Creating training corpora for nlg micro-planning. In *55th annual meeting of the Association for Computational Linguistics (ACL)*.

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.

Sarthak Garg, Stephan Peitz, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Jointly Learning to Align and Translate with Transformer Models. In *2019 Conference on Empirical Methods in Natural Language Processing (EMNLP) and 9th International Joint Conference on Natural Language Processing (IJCNLP)*, Hong Kong.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Albert Gatt and Ehud Reiter. 2009. Simplenlg: A realisation engine for practical applications. In *Proceedings of the 12th European Workshop on Natural Language Generation (ENLG 2009)*, pages 90–93.

Sebastian Gehrmann, Tosin Adewumi, Karmanya Aggarwal, Pawan Sasanka Ammanamanchi, Aremu Anuoluwapo, Antoine Bosselut, Khyathi Raghavi Chandu, Miruna Clinciu, Dipanjan Das, Kaustubh D Dhole, et al. 2021. The GEM benchmark: Natural language generation, its evaluation and metrics. *arXiv preprint arXiv:2102.01672*.

Sebastian Gehrmann, Falcon Dai, Henry Elder, and Alexander Rush. 2018. End-to-end content and plan selection for data-to-text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 46–56, Tilburg University, The Netherlands. Association for Computational Linguistics.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 946–958.

Eli Goldberg, Norbert Driedger, and Richard I Kittredge. 1994. Using natural-language processing to produce weather forecasts. *IEEE Expert*, 9(2):45–53.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Hamza Harkous, Isabel Groves, and Amir Saffari. 2020. Have your text and use it too! end-to-end neural data-to-text generation with semantic fidelity. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2410–2424.

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

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339.

Jena D Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2020. Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs. *arXiv preprint arXiv:2010.05953*.

Peter Izsak, Moshe Berchansky, and Omer Levy. 2021. How to train bert with an academic budget. *arXiv preprint arXiv:2104.07705*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mihir Kale and Abhinav Rastogi. 2020a. Template guided text generation for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6505–6520, Online. Association for Computational Linguistics.

Mihir Kale and Abhinav Rastogi. 2020b. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

Zdeněk Kasner and Ondřej Dušek. 2020a. Data-to-text generation with iterative text editing. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 60–67.

Zdeněk Kasner and Ondřej Dušek. 2020b. Train hard, finetune easy: Multilingual denoising for rdf-to-text generation. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 171–176.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *CoRR*, abs/1909.05858.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686.

Anirban Laha, Parag Jain, Abhijit Mishra, and Karthik Sankaranarayanan. 2019. Scalable micro-planned generation of discourse from structured data. *Computational Linguistics*, 45(4):737–763.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213, Austin, Texas. Association for Computational Linguistics.

Chris van der Lee, Chris Emmery, Sander Wubben, and Emiel Krahmer. 2020. The cacapo dataset: A multilingual, multi-domain dataset for neural pipeline and end-to-end data-to-text generation. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 68–79.

Yuanmin Len, François Portet, Cyril Labbé, and Raheel Qader. 2020. Controllable neural natural language generation: comparison of state-of-the-art control strategies. In *WebNLG+: 3rd Workshop on Natural Language Generation from the Semantic Web*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Peiguang Li, Hongfeng Yu, Wenkai Zhang, Guangluan Xu, and Xian Sun. 2020. Sa-nli: A supervised attention based framework for natural language inference. *Neurocomputing*, 407:72–82.

Tomasz Limisiewicz, David Mareček, and Rudolf Rosa. 2020. Universal dependencies according to bert: both more specific and more general. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2710–2722.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020a. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020b. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

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, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421. The Association for Computational Linguistics.

Jonathan Mallinson, Aliaksei Severyn, Eric Malmi, and Guillermo Garrido. 2020. FELIX: Flexible text editing through tagging and insertion. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1244–1255, Online. Association for Computational Linguistics.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5054–5065.

Christopher Manning and Hinrich Schutze. 1999. *Foundations of statistical natural language processing*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448.

Hongyuan Mei, Mohit Bansal, and Matthew R Walter. 2016. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 720–730.

Merriam-Webster. 2021. Domain.

Simon Mille, Stamatia Dasiopoulou, Beatriz Fisas, and Leo Wanner. 2019. Teaching forge to verbalize dbpedia properties in spanish. In *Proceedings of the*

12th International Conference on Natural Language Generation, pages 473–483.

Martin Molina, Amanda Stent, and Enrique Parodi. 2011. Generating automated news to explain the meaning of sensor data. In *International Symposium on Intelligent Data Analysis*, pages 282–293. Springer.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-step: Separating planning from realization in neural data-to-text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2267–2277.

Shashi Narayan, Claire Gardent, Shay B Cohen, and Anastasia Shimorina. 2017. Split and rephrase. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 606–616.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Comput. Linguist.*, 29(1):19–51.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. Gector–grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170.

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*, pages 311–318.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Matthew E Peters, Sebastian Ruder, and Noah A Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473.

François Portet, Ehud Reiter, Albert Gatt, Jim Hunter, Somayajulu Sripada, Yvonne Freer, and Cindy Sykes. 2009. Automatic generation of textual summaries from neonatal intensive care data. *Artificial Intelligence*, 173(7-8):789–816.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, pages 1–26.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Clément Rebuffel, Marco Roberti, Laure Soulier, Geoffrey Scoutheeten, Rossella Cancelliere, and Patrick Gallinari. 2021. Controlling hallucinations at word level in data-to-text generation. *arXiv preprint arXiv:2102.02810*.

Nils Reimers, Iryna Gurevych, Nils Reimers, Iryna Gurevych, Nandan Thakur, Nils Reimers, Johannes Daxenberger, Iryna Gurevych, Nils Reimers, Iryna Gurevych, et al. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Ehud Reiter. 2019. Natural language generation challenges for explainable AI. In *Proceedings of the 1st Workshop on Interactive Natural Language Technology for Explainable Artificial Intelligence (NL4XAI 2019)*, pages 3–7.

Ehud Reiter and Robert Dale. 1997. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. Studies in Natural Language Processing. Cambridge University Press.

Ehud Reiter, Somayajulu Sripada, Jim Hunter, Jin Yu, and Ian Davy. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167(1-2):137–169.

Ehud Reiter and Craig Thomson. 2020. Shared task on evaluating accuracy. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 227–231.

Leonardo FR Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2020. Investigating pretrained language models for graph-to-text generation. *arXiv preprint arXiv:2007.08426*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Rudolf Rosa, Tomáš Musil, Ondřej Dušek, Dominik Jurko, Patrícia Schmidtová, David Mareček, Ondřej Bojar, Tom Kocmi, Daniel Hrbek, David Košťák, et al. 2021. Theaitre 1.0: Interactive generation of theatre play scripts. *arXiv preprint arXiv:2102.08892*.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

Jiirgen Schmidhuber. 1990. Making the world differentiable: On using self-supervised fully recurrent n eu al networks for dynamic reinforcement learning and planning in non-stationary environm nts.

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, Vancouver, Canada. Association for Computational Linguistics.

Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D. Manning. 2019. Do massively pretrained language models make better storytellers? In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 843–861, Hong Kong, China. Association for Computational Linguistics.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Robert Speer, Catherine Havasi, J CHAIDEZ, J VENEZUELA, and Y KUO. 2012. Conceptnet 5. *Tiny Transactions of Computer Science*.

Felix Stahlberg and Shankar Kumar. 2020. Seq2edits: Sequence transduction using span-level edit operations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5147–5159.

Shang-Yu Su, Chao-Wei Huang, and Yun-Nung Chen. 2020. Towards unsupervised language understanding and generation by joint dual learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 671–680. Association for Computational Linguistics.

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.

André Luiz Rosa Teixeira, João Campos, Rossana Cunha, Thiago Castro Ferreira, Adriana Pagano, and Fabio Cozman. 2020. Damata: A robot-journalist covering the brazilian amazon deforestation. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 103–106.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2019. Generating token-level explanations for natural language inference. In *Proceedings of NAACL-HLT*, pages 963–969.

Bayu Trisedya, Jianzhong Qi, and Rui Zhang. 2020. Sentence generation for entity description with content-plan attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9057–9064.

Shikhar Vashishth, Shyam Upadhyay, Gaurav Singh Tomar, and Manaal Faruqui. 2019. Attention interpretability across nlp tasks. *arXiv preprint arXiv:1909.11218*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Superglue: a stickier benchmark for general-purpose language understanding systems. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 3266–3280.

Tsung-Hsien Wen, Milica Gašić, 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, Lisbon, Portugal. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498.

Zixiaofan Yang, Arash Einolghozati, Hakan Inan, Keith Diedrick, Angela Fan, Pinar Donmez, and Sonal Gupta. 2020. Improving text-to-text pre-trained models for the graph-to-text task. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 107–116.

Thomas Zenkel, Joern Wuebker, and John DeNero. 2020. End-to-End Neural Word Alignment Outperforms GIZA++. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1605–1617, Online. Association for Computational Linguistics.

Chao Zhao, Marilyn Walker, and Snigdha Chaturvedi. 2020. Bridging the structural gap between encoding and decoding for data-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2481–2491, Online. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27.