Have Your Text and Use It Too!
End-to-End Neural Data-to-Text Generation with Semantic Fidelity

Hamza Harkous
Amazon Alexa *
hamza.harkous@gmail.com

Isabel Groves
Amazon Alexa
isabeg@amazon.co.uk

Amir Saffari
Amazon Alexa
amsafari@amazon.co.uk

Abstract

End-to-end neural data-to-text (D2T) generation has recently emerged as an alternative to pipeline-based architectures. However, it has faced challenges in generalizing to new domains and generating semantically consistent text. In this work, we present DATATUNER, a neural, end-to-end data-to-text generation system that makes minimal assumptions about the data representation and the target domain. We take a two-stage generation-reranking approach, combining a fine-tuned language model with a semantic fidelity classifier. Each of our components is learnt end-to-end without the need for dataset-specific heuristics, entity delexicalization, or post-processing. We show that DATATUNER achieves state of the art results on the automated metrics across four major D2T datasets (LDC2017T10, WebNLG, ViGGO, and Cleaned E2E), with a fluency assessed by human annotators nearing or exceeding the human-written reference texts. We further demonstrate that the model-based semantic fidelity scorer in DATATUNER is a better assessment tool compared to traditional, heuristic-based measures. Our generated text has a significantly better semantic fidelity than the state of the art across all four datasets.

1 Introduction

Data-to-Text generation (D2T) is defined as automatically generating natural language texts from non-linguistic inputs (Reiter and Dale, 2000). Interest in this task has been driven by its applicability to specialized domains. For instance, D2T has been applied generating weather reports (Liang et al., 2009), restaurant descriptions (Novikova et al., 2017b), and video game dialogues (Juraska et al., 2019). Recently, researchers have investigated D2T with more diverse domains to arrive at more generalizable text generation (such as works on the LDC2017T10 (Knight et al., 2017) and WebNLG (Gardent et al., 2017) datasets).

Traditional approaches to D2T follow a pipeline-based methodology, dividing the problem into several sub-problems (Reiter and Dale, 2000; Gatt and Krahmer, 2018). These include content selection (which information to include in the text), text structuring (the order in which to present the data), sentence aggregation (which information goes in individual sentences), lexicalization (finding the right words and phrases to express the data), referring expression generation (selecting the words and phrases to identify domain objects), and linguistic realization (combining all the generated words and phrases into well-formed sentences).

In recent years, there has been growing interest in going beyond pipeline-based approaches towards end-to-end (E-to-E) methods driven by recent advancements in deep learning (Lebret et al., 2016; Novikova et al., 2017b; Castro Ferreira et al., 2019; Dušek et al., 2020). Such methods can be trained with (*data,text*) tuples that can be efficiently collected at scale. In contrast, in pipeline approaches, each step requires its own setup and training data, such as semantic alignments between sections of the text and components of the meaning representation. This makes them more costly and complex to develop and more prone to error propagation.

To date, end-to-end D2T has faced two main challenges: (1) **generalization** to unseen domains and (2) maintaining **semantic fidelity** to accurately convey the source data. In a recent comparative study, Castro Ferreira et al. (2019) found that, compared to the best pipeline-based system, E-to-E approaches based on GRU and Transformer architectures scored more than 35 BLEU points lower on unseen domains from the WebNLG dataset. Moreover, E-to-E systems scored worst for semantic accuracy.

To address these challenges, we introduce DATATUNER, an E-to-E, domain-independent D2T system that makes no assumptions about the generated text or meaning representation. At its core, DATATUNER leverages a pretrained language model with fine-grained state embeddings to

* Work done while at Amazon; currently affiliated with Google

achieve strong generalization. It also employs a weakly-supervised Semantic Fidelity Classifier (SFC) to detect and avoid generation errors (such as hallucination, omission, repetition, and value errors). We further repurpose this classifier to assess the outputs of any D2T system, overcoming the limitations of existing heuristic-heavy methods for detecting semantic errors.

In this work, we deliver three main contributions across four major D2T datasets from various domains and meaning representations:

- We show that DATATUNER pushes the state of the art on automated metrics by significant margins, ranging from 1.2 to 5.9 BLEU points, compared to the best existing pipeline-based and E-to-E techniques.
- With a crowdsourcing experiment on Amazon Mechanical Turk, we demonstrate that DATATUNER generates text with significantly better fluency than existing works. On two datasets, our texts are even judged to be better, on-average, than the human references.
- We show that our model-based semantic accuracy metric is 4.2% to 14.2% more accurate in detecting semantic errors than existing heuristic-based approaches. As a result, DATATUNER significantly improves the semantic accuracy of generated text as assessed by manual annotation.

2 Related Work

Pipeline vs. End-to-End Approaches: Within the pipeline-based paradigm, several studies have illustrated that breaking the D2T problem into sub-problems improves overall performance. Moryossef et al. (2019b) showed that separating planning from realization helps achieve better semantic faithfulness compared to an E-to-E neural approach on the WebNLG dataset. Castro Ferreira et al. (2019) conducted a comparative study across a variety of E-to-E and pipeline approaches with WebNLG. They concluded that the latter are significantly better at generalizing to unseen domains. However, so far, the E-to-E approaches in these studies have been trained from scratch on the task dataset. Our work investigates whether using a pretrained model with strong language understanding and generation capabilities raises the performance of E-to-E models.

Structured Representations of the Data Another thread of research focuses on developing better encoders for meaning representation languages, exploiting their structural properties. This is particularly relevant to AMR (Damonte and Cohen, 2019; Ribeiro et al., 2019; Zhu et al., 2019; Guo et al., 2019). Damonte and Cohen (2019) showed that replacing sequential encoders with a graph encoder improves text quality as measured by BLEU and METEOR scores. Zhu et al. (2019) proposed using self-attention to better model the relations between indirectly connected AMR components. Our work differs in that it does not require any explicit assumption about the structure of the meaning representation or the relations between its components.

Semantic Fidelity Guarantees To improve semantic fidelity (how accurately the generated text conveys the meaning) in E-to-E architectures, one approach has been to train reverse "Text-to-Data" models (Chisholm et al., 2017; Agarwal et al., 2018). We take a different approach in this work as we are focused on semantically verifying the generated outputs. We aim to build a semantic fidelity model that can generalize better by not having to learn to convert unseen values or entities to their corresponding representation in the data. Another approach has been to rely on heuristics that map data values to potential realizations in the text, thus computing a Slot Error Rate (SER) metric (Dušek et al., 2019; Juraska et al., 2019; Moryossef et al., 2019a). For instance, Dušek et al. (2019) use SER for reranking beam elements during decoding from an attention-based sequence-to-sequence model on the Cleaned E2E dataset. Juraska et al. (2019) used it similarly with a transformer model on the ViGGO dataset. This technique, despite aiming for more transparency, is difficult to scale to wider domains. Moreover, for meaning representations which are not dominated by named entities, designing the rules to ensure consistency becomes more difficult.

3 Problem Description

To illustrate our approach to the D2T task and motivate the architecture choices, we start by formalizing the task and describing the datasets used in our study.

3.1 Data-to-Text Task

The D2T task is defined as generating text T from data D that is encoded via a meaning representation *MR*. We assume that content selection is done prior to the D2T task, an assumption also made in the datasets we use. Therefore, the text T should have *semantic fidelity* by conveying all the input data, and only the input data.

3.2 Datasets

We selected the major datasets that satisfy the task definition above. Each dataset consists of (D,T) pairs. The following briefly describes each dataset with examples of how the data is preprocessed/linearized ready to be fed into the models. Note that we rely on adding special tokens (highlighted in bold below) during preprocessing to better guide our models.

3.2.1 WebNLG

For WebNLG data, D is a set of 1 to 7 DBPedia triples and T is an English text verbalizing these (Gardent et al., 2017). The test data spans 15 different domains, 10 of which appear in the training data. For data linearization, we concatenate the triples, adding special tokens for 'subject', 'predicate', and 'object' indicators. We convert camel- and snake-case strings to sentence-case. For fair comparison with the state of the art, we use v1.4 from Castro Ferreira et al. (2018).

Example 3.1

D= Subject: Aarhus | Predicate: leaderName | Object: Jacob_Bundsgaard
Linearized D= <**subject**> Aarhus <**predicate**> leader name <**object**> Jacob Bundsgaard
T= The leader of Aarhus is Jacob Bundsgaard.

3.2.2 LDC2017T10

In the LDC2017T10 dataset (Knight et al., 2017), D is an Abstract Meaning Representation (AMR) graph representing "who is doing what to whom" for each sentence in T. The texts include broadcast conversations, newswire and weblogs. We linearized using the preprocessing script by Ribeiro et al. (2019), without lowercasing. We merged multiple leaves that correspond to one entity (e.g., "*United States*" below) and replaced each role specifier (words starting with a colon, such as "*:name*") with a special token.

Example 3.2

D= (r / respond−01
:ARG0 (c / country :wiki ''United_States''
:name (n / name :op1 ''United''
:op2 ''States''))
:ARG1 (d / develop−01
:mod (t / that))
:ARG2 (c2 / condemn−01
:manner (s / swift)))
Linearized D= (respond <:**ARG0**>
(country <:**name**> (United States))
<:**ARG1**> (develop <:**mod**> (that))
<:**ARG2**> (condemn <:**manner**> (swift)))
T= The United States responded to that development with swift condemnation.

3.2.3 Cleaned E2E

The Cleaned E2E dataset recently introduced in (Dušek et al., 2019) is an automatically cleaned version of the original E2E dataset (Novikova et al., 2017b), aiming to eliminate omissions and hallucinations in the human text by fixing the corresponding *MR*. Each *MR* consists of 3 to 8 slot-value pairs in the restaurant domain. We preprocessed D by adding special tokens before each slot type.

Example 3.3

D= name[Zizzi], eatType[coffee shop], area[riverside]
Linearized D= <**name**> name=[Zizzi]; <**eatType**> eatType=[coffee shop]; <**area**> area=[riverside]
T= You can find a coffee shop named Zizzi in the riverside area.

3.2.4 ViGGO

In the ViGGO dataset (Juraska et al., 2019), D is a meaning representation with one of 9 dialogue acts (e.g. *give_opinion, confirm, suggest*, etc.) and 1 to 8 slot-value pairs from 14 different video game attributes (e.g. NAME, GENRES, etc.). Each T is an utterance representing a dialogue turn in the video games domain. In preprocessing, we add special tokens at the beginning and end, representing the dialog act, and special tokens before each slot type.

Example 3.4

D= request(developer[EA Canada], specifier[favorite])
Linearized D= <**request**> request (<**developer**> developer: [EA Canada], <**specifier**> specifier: [favorite] <**request**>)
T= What's your favorite game that EA Canada has made?

3.3 Datasets Discussion

The datasets vary widely. LDC2017T10 dataset is not bounded to specific domains. Hence, although the AMR format closely describes the text, it is non-trivial to generalize from the training to test data. WebNLG covers a wide, but restricted set of domains, only a subset of which are present in the training data. However it has high lexical diversity. The number of unique words in the test set of WebNLG is 7253 (63% of them capitalized), compared to 5533 (21.6% capitalized) for LDC2017T10, 2014 (33% capitalized) for ViGGO, and 1966 (29% capitalized) for Cleaned E2E. Measured with the New DaleChall readability score (Dale and Chall, 1948), LDC2017T10 had the highest difficulty score (6.49) compared to 1.03, 0.85, and 1.02 for the WebNLG, Cleaned E2E, and ViGGO datasets respectively. In terms of quality, ViGGO has been designed with

the goal of perfect semantic fidelity, and Cleaned E2E was heavily filtered from the original dataset to achieve that. On the other hand, the other datasets' versions we use have not undergone such filtering.

4 DATATUNER Architecture

Given the diverse meaning representations we tackle, we designed DATATUNER to be highly generic, allowing D2T generators to be built for new datasets with minimal work beyond data preprocessing. At a high-level, our text generation system takes a 2-stage approach through *generation* and *reranking*. First, we fine-tune a pretrained language model on the D2T task using the task's training data. Next, we build a specialized semantic fidelity classifier trained on an automatically-generated, task-specific corpus. Using these models, we construct a customized beam-search decoder that ranks candidates based on the probabilities from the language model, and, at its final stage, reranks them based on the classifier's labels.

4.1 Data-to-Text Model Fine-tuning

The first component in DATATUNER is the fine-tuned Data-to-Text Language Model (D2T-LM). We build on the pretrained OpenAI GPT-2 model (Radford et al., 2019), a multi-layer, autoregressive language model. Each layer is a transformer decoder block (Vaswani et al., 2017) of masked multi-headed attention and a fully connected layer. We provide a full diagram of the model operation in Figure 2 of the Appendix.

Inputs: To create the input, we concatenate the data D and the text T into a single sequence ($<data>\{D\}<text>\{T\}$). The tokens $<data>$ and $<text>$ are special tokens appended to GPT-2's original vocabulary; their embeddings are learnt during fine-tuning. In addition, we append to the vocabulary the task-dependent special tokens described above.After tokenization, we get a sequence of subword tokens, which are encoded to point to vocabulary indices:

$$S = (<data>, d_1, ...d_k, <text>, t_1, ...t_m) = (s_0, ...s_n)$$

GPT-2 additionally expects positional encodings that help it capture the input tokens' order. We also add a third type of input: state embeddings. These are analogous to the "Segment Embeddings", introduced in BERT (Devlin et al., 2019) to distinguish between sentence pairs in the next sentence prediction task. They have also been used by Wolf et al. (2019b) to differentiate between different speakers' utterances in dialogue. We apply these state embeddings at a more fine-grained level to give the model a hint on the type of the data being handled. The state vector for S is a vector of tokens with size $|S|$, with each token ID indicating the type of s_i. We use a simple rule for all datasets: the state token ID of any token s_i is the ID of the last special token preceding it (i.e. in the range $(s_0...s_i)$ inclusively). One interesting feature of GPT-2 is its use of Byte-Pair Encoding (BPE) (Sennrich et al., 2016) on bytes instead of unicode characters. Hence, with a modestly-sized subword vocabulary of around 50K, it can encode any input text and score any output sequence, without suffering from unknown tokens. This is beneficial for our task where named entities are common.

Training: The input embeddings, positional embeddings, and state embeddings are added together and fed to the first GPT-2 layer. The last GPT-2 layer output is then normalized using "LayerNorm" (Ba et al., 2016) before passing it to a linear layer added on top. The weights of the latter are tied to the input embeddings. Finally, a softmax is applied to the output of the linear layer to generate probability distributions of the output tokens. Our training objective is a language modeling one where we aim to find the set of weights θ that minimize the cross-entropy loss

$$\ell = \sum_{i=|D|+2}^{|S|} \log P_\theta(s_i | s_0, ...s_{i-1}) \quad (1)$$

Note that, since our task is to generate text given the data, we mask the data component in the loss above, and sum the loss from index $|D|+2$ (i.e., after the $<text>$ token). We use *AdamW* as an optimizer (Loshchilov and Hutter, 2019).

4.2 Semantic Fidelity Classifier

The second component of DATATUNER is the Semantic Fidelity Classifier (SFC). A text is deemed to possess semantic fidelity if it accurately conveys all parts of the input data without omitting any nor adding additional data. This component provides an additional assessment of how accurately the generated text reflects the input data. Our approach draws parallels between this task and natural language inference (NLI) tasks, where the goal is to determine whether a "hypothesis" is true, false, or undetermined given a "premise". Similarly, in semantic fidelity classification, we aim to determine if the text is "accurate" or contains some "omission", "repetition", "hallucination", or "value errors". We build on the success seen by pretrained models such

as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) for NLI, and cast the problem as a sentence-pair classification task for the (*Data, Text*) pairs, using RoBERTa as a base encoder.

Training Data Generation: The classifier's training data should consist of semantically faithful and semantically incorrect examples. When evaluating consistency in abstractive summarization, weakly supervised models trained on domain-specific data have been shown to outperform supervised models trained on out-of-domain, human-annotated data (Kryściński et al., 2019). Motivated by that, we generate the training data for the SFC automatically from training data for the main D2T task. We define a set of simple, dataset-independent transformations that account for common errors in data-to-text generation. For each tuple (D_i,T_i) in the training data, we split the text T_i into sentences, using the Spacy sentence tokenizer (Honnibal and Montani, 2017). Then we generate the following variations:

- **Accurate**: This is the text T_i.
- **Omission**: generated by removing the shortest sentence in T_i (to help detect subtle omissions).
- **Repetition**: generated by taking a random sentence in the text T_i and inserting it before a random other sentence in T_i.
- **Hallucination**: generated by selecting a random sentence from another training text $T_{j\neq i}$ and inserting it before a random sentence in T_i.
- **Value Errors**: generated by selecting a random value x that occurs verbatim in both D_i and T_i. We replace x in T_i with a random other value from D_i. For slot-based *MR* (Cleaned E2E and ViGGO), x is selected from the slots' values. For graph-based *MR* (LDC2017T10), x is selected from the graph's leaves. For RDF triples (WebNLG dataset), x is chosen from the triples' subjects and objects.

For each tuple (D_i,T_i) from the original dataset, we get a set of new tuples for the SFC, consisting of (D_i, T_j, l) for each error label l above and (D_i,T_i,"*accurate*") for the accurate label.

Model Input: As shown in Figure 1, we concatenate the data and text tokens, adding the special start (<*s*>) and end (</*s*>) tokens used during the training of RoBERTa. In addition to subword token embeddings, we add positional embeddings (representing the position in the input) and segment embeddings (representing the data type vs. the text type).

Training: The 3 embeddings are summed element-wise to produce the input representation passed to RoBERTa's first encoder layer. Each layer subsequently applies a self-attention followed with a feed-forward network. Similar to the handling of classification problems in BERT and RoBERTa, we take the output hidden layer corresponding to the very first token (<*s*>) and pass that through an additional single-layer neural network. The model is trained as a multi-class classifier (5 labels), with a cross-entropy loss as the objective and AdamW as the optimizer.

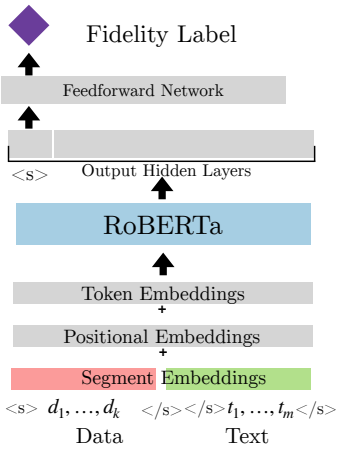

Figure 1: Semantic fidelity classifier setup

4.3 Decoder

Our decoding algorithm for the D2T-LM is based on beam-search. At each decoding step, items are ranked according to the score R below, which multiplies the conditional probabilities' product with a length normalization factor. Low-scoring candidates are dropped once the number of candidates exceeds the beam size.

$$R = \frac{1}{(i-(|D|+2))^{\alpha}} \prod_{|D|+2}^{i} P(s_i|s_0 \ldots s_{i-1})$$

Compared to traditional beam search, we do not aggregate probabilities from the start of the sequence, but from the start of the text component (index $|D|+2$). Moreover, the length normalization is adjusted to only account for the text component.We do this because we fine-tuned the D2T-LM on generating text given data as a context, and not on generating the data itself. Hence, we remove the data tokens from the beam-scoring function to prevent the decoder from favoring longer sequences. In our experiment, we use a value of $\alpha = 0.75$. At the end of the beam-search, we use the SFC to rerank the *complete* candidates (terminated with

an end-of-sequence token) in the beam. For the reranking metric, we use the following binary score:

$$\mathbb{1}_{SFC(D_i,T_i)=\text{“}accurate\text{”}}$$

Hence, we push the text T_i to the top of the beam if our SFC labels the (D_i,T_i) tuple as “*accurate*”. We resolve ties using the original D2T-LM scores. An alternative strategy would have been to apply the reranking at each decoding stage, but we empirically found that strategy to have negligible gains in terms of the “accurate” beam outputs while requiring a cost that grows with the text size. In addition to helping surface semantically accurate outputs, the SFC labels can be used to assess whether the generated text is usable in practice. In our experiments, we compare this model-based approach to the current heuristic-heavy approach commonly used.

5 Experiments

For each dataset, we generate the outputs from three versions of DATATUNER. DATATUNER_NO_FC/FS simply relies on the D2T-LM, with no SFC-based reranking and a coarse-grained version of the state embeddings that contains only <*data*> and <*text*> tokens (as done by Wolf et al. (2019b)). DATATUNER_NO_FC adds the fine-grained state embeddings described in Section 4.1 to DATATUNER_NO_FC/FS. The third variant, DATATUNER_FC, additionally includes the SFC-based reranking. For the SFC, we generate the synthetic dataset and train the model using the RoBERTa-large model (355M parameters) on lower-cased text. On the synthetic test set, the weakly-supervised classifier has a macro-averaged F1-score (across 5 classes) of 97%, 97%, 98%, and 98% for the LDC2017T10, WebNLG, Cleaned E2E, and ViGGO datasets respectively.We use the models bundled within the HuggingFace Transformers library (Wolf et al., 2019a). For the D2T-LM, we select the GPT-2-Medium model (with 345M-parameters) as the base model and set the beam search width during decoding to 5. All our experiments were performed on a single machine with Nvidia Tesla v100 16GB GPUs.

We evaluate each variant's outputs with automated metrics, crowdsourced fluency evaluation, and expert-annotated semantic assessment. We also quantify the efficacy of our semantic fidelity classifier. We compare against the state of the art systems on each dataset, selected based on BLEU scores. In the supplementary material, we include the outputs from our system variants as well as the main training hyperparameters.

D	Model	B	M	R	C
LDC2017T10	DATATUNER_FC	**37.7**	**38.9**	**65.1**	**3.9**
	DATATUNER_NO_FC	37.2	38.4	65.0	**3.9**
	DATATUNER_NO_FC/FS	35.6	37.3	64.4	3.8
	Zhu et al. (2019)	31.8	36.4	-	-
	Guo et al. (2019)	30.4	-	-	-
	Ribeiro et al. (2019)	27.9	33.2	-	-
WebNLG	DATATUNER_FC	52.4	**42.4**	**66.0**	**3.7**
	DATATUNER_NO_FC	**52.9**	41.9	65.9	**3.7**
	DATATUNER_NO_FC/FS	51.6	40.6	64.9	3.6
	Castro Ferreira et al. (2019) Pipe.	51.7	32.0	-	-
	Castro Ferreira et al. (2019) E2E	33.5	25.0	-	-
	Moryossef et al. (2019b) Pipe.	47.4	39.1	63.1	2.7
Cleaned E2E	DATATUNER_FC	**43.6**	**39.0**	57.5	**2.0**
	DATATUNER_NO_FC	**43.6**	**39.0**	57.5	**2.0**
	DATATUNER_NO_FC/FS	43.3	38.9	**57.6**	**2.0**
	Dušek et al. (2019) (TGen+)	40.5	37.6	56.0	1.8
ViGGO	DATATUNER_FC	**53.6**	**39.4**	**64.0**	**2.7**
	DATATUNER_NO_FC	53.4	39.1	63.8	**2.7**
	DATATUNER_NO_FC/FS	51.4	38.9	62.7	2.5
	Juraska et al. (2019)	52.1	39.1	63.8	2.5

Table 1: Evaluation of the different systems based on automated metrics.

5.1 Automated Evaluation

For each test set, we compute BLEU (B) (Papineni et al., 2002), which measures the n-gram precision, METEOR (M) (Lavie and Agarwal, 2007), which is based on the harmonic mean of the unigram precision and recall while accounting for stem and synonymy matching, ROUGE$_L$ (R) (Lin, 2004), which calculates the recall for the longest common subsequence, and CIDer (C) (Vedantam et al., 2015), which is based on the TF-IDF scoring of the n-grams. We used the official evaluation scripts of the E2E challenge. [1]. Table 1 compares the results generated by DATATUNER variants against the state of the art on each dataset.

Improvements from the D2T-LM alone: Analyzing the simple DATATUNER_NO_FC/FS model compared to the state of the art on each dataset, we find that it already improves the BLEU score across 2 datasets and the METEOR score across 3 datasets. This indicates that the D2T-LM component of DATATUNER is itself contributing to achieving an end-to-end state of the art system without needing any delexicalization or MR-specific encoding.

Fine-grained state embeddings matter: We notice a consistent trend across the 4 datasets: adding fine-grained state embeddings boosts the classifier's performance on these metrics, for

[1] https://github.com/tuetschek/e2e-metrics

instance, from 0.3 (on Cleaned E2E) to 2.0 BLEU points (on ViGGO).

SFC effect on automated metrics: Several studies highlighted the shortcomings of automated metrics in evaluating semantic adequacy (Novikova et al., 2017a; Shimorina, 2018). Along these lines, compared to our DATATUNER_NO_FC model, we observe slight additional boosts from introducing the SFC classifier with the DATATUNER_FC variant. Interestingly, DATATUNER_FC always has the highest METEOR score, which was the only metric found by Shimorina (2018) to be correlated with semantic adequacy.

Largest boost on the most complex text: DATATUNER had the widest improvement of 5.9 additional BLEU points on the LDC2017T10 dataset. This is interesting, given that (1) the text in LDC2017T10 is typically long with more complex sentence structures (cf. Section 3.3) and that (2) the baseline systems targeting AMR-to-text (Zhu et al., 2019; Guo et al., 2019; Ribeiro et al., 2019) built more sophisticated architectures compared to other datasets (e.g.,ViGGO and Cleaned E2E). This illustrates our system's ability to work across a spectrum of data representations and text complexity.

6 Human Evaluation of Fluency

We conduct human evaluation of fluency, also known as naturalness or readability for 150 examples sampled at random from each dataset. We sourced the state of the art systems' outputs either from the paper's repository (WebNLG) or directly from the authors (LDC2017T10, ViGGO, Cleaned E2E). For fluency, we use Amazon's Mechanical Turk to ask crowd workers to indicate how fluent a text is on a 7-point Likert scale using sliders, where "high fluency" is defined as "grammatical, natural, and could have been produced by a skilled native speaker". Following findings from (Novikova et al., 2018; Van Der Lee et al., 2019) for acquiring more consistent human ratings, texts from different systems generated for the same meaning representation are presented in every task for annotators to score them relative to each other. We also include the human-written text, and randomize the texts' order. For a fair comparison, we lower-case our generated texts for the LDC2017T10 dataset to match the outputs of Zhu et al. (2019). We also detokenize outputs from that work to avoid these biasing the workers. We choose experienced annotators (completed >500 tasks) with high previous performance (>97% of previous tasks accepted) from the USA.

D	Model	F	DSA	Q_D	HSA	Q_H
LDC2017T10	DATATUNER_FC	$4.79^{s,h}$	$81.8^{s,h}$		–	
	DATATUNER_NO_FC	4.87^{s}	$70.5^{s,h}$		–	
	DATATUNER_NO_FC/FS	$4.78^{s,h}$	$65.8^{s,h}$	90.8	–	-
	Zhu et al. (2019)	3.97^{h}	62.4^{h}		–	
	Human	5.05	93.1		–	
WebNLG	DATATUNER_FC	5.23^{s}	$91.7^{s,h}$		$58.1^{s,h}$	
	DATATUNER_NO_FC	5.20^{s}	$81.4^{s,h}$		$54.1^{s,h}$	
	DATATUNER_NO_FC/FS	5.27^{s}	$73.6^{s,h}$	**87.5**	$43.6^{s,h}$	73.3
	Castro Ferreira et al. (2019)	4.54^{h}	50.5^{h}		33.7^{h}	
	Human	5.21	94.7		41.2	
Cleaned E2E	DATATUNER_FC	$5.46^{s,h}$	$99.3^{s,h}$		$97.3^{s,h}$	
	DATATUNER_NO_FC	$5.46^{s,h}$	99.0^{h}		$97.1^{s,h}$	
	DATATUNER_NO_FC/FS	$5.45^{s,h}$	98.9^{h}	**89.2**	$97.1^{s,h}$	75.0
	Dušek et al. (2019) TGen+	5.23^{h}	98.9^{h}		98.0^{h}	
	Human	4.42	99.9		100.0	
ViGGO	DATATUNER_FC	$5.77^{s,h}$	97.2^{s}		$74.5^{s,h}$	
	DATATUNER_NO_FC	$5.76^{s,h}$	92.8^{h}		$82.3^{s,h}$	
	DATATUNER_NO_FC/FS	5.60^{h}	91.7^{h}	92.5	$82.5^{s,h}$	88.3
	Juraska et al. (2019)	5.58^{h}	92.8^{h}		90.9	
	Human	5.34	97.1		91.9	

Table 2: Human evaluation of the fluency (F), DATATUNER Semantic Accuracy (DSA), Heuristic Semantic Accuracy (HSA), and quality measures Q_D and Q_H for DSA and HSA. The superscripts s and h imply a statistically significant difference compared to the state of the art and the human baseline respectively.)

Improvement on the state of the art: As shown in Table 2, compared to the human baseline, our DATATUNER_FC model improves the fluency on all four datasets compared to the state of the art systems with statistically significant margins ($p < 0.05$). For computing significance measures, we use the pairwise Wilcoxon signed-rank test (Wilcoxon, 1992) with the null hypothesis that the fluency values for each pair of systems come from the same distribution. For LDC2017T10, where DATATUNER_FC had the largest gap in BLEU score (+5.9), we observe the widest fluency improvement (+0.82) compared to Zhu et al. (2019). Interestingly, despite the fact that DATATUNER_FC scored 0.7 higher on BLEU compared to the pipeline approach in (Castro Ferreira et al., 2019) for WebNLG, the difference in fluency is 0.69. We conjecture that this originates from two main sources. First, semantic errors in the outputs might be perceived by annotators as breaking the sentence fluency. For example, one text contained the phrase "*has a runway length of Shehbaz Sharif*". Second, the pipeline approach had a sizeable portion of non-realized outputs (e.g. "PATIENT-1 is made with PATIENT-1 and PATIENT-2."), which were annotated as non-fluent. On the closed-domain datasets (ViGGO and Cleaned E2E), we notice that the fluency margins shrink while still being statistically significant. This is expected as these datasets have a narrow set of

sentence formulations that are easier to learn.

Improvement on the human baseline: Surprisingly, we find that DATATUNER_FC has received a higher overall average fluency score on 3 out of the 4 datasets compared to the human baseline. This difference is statistically significant in both Cleaned E2E and ViGGO, with the largest difference being 1.04 points for the Cleaned E2E. Investigating, we found several low-scored texts had an informal style and problems in sentence construction. One example contained "*It serves Chinese food for less.*" One explanation could be that, once fine-tuned on a large enough dataset, our models have less tendency to deviate from common formulations that are favored by annotators.

7 Human Evaluation of Semantic Fidelity

For assessing semantic accuracy, we compare two approaches. The first uses heuristics to label each data-text tuple as accurate (A_H) or erroneous (E_H). For this, we use the heuristics by Shimorina and Gardent (2018) for WebNLG, by Juraska et al. (2018) for ViGGO, and by Dušek et al. (2019) for Cleaned E2E. We are not aware of heuristic-based scripts for LDC2017T10. Then we compute Heuristic Semantic Accuracy (HSA) of a dataset as the fraction with the label A_H. The second approach uses the SFC component in DATATUNER to assign accurate (A_D) or erroneous (E_D) labels for each data-text tuple. We compute DATATUNER Semantic Accuracy (DSA) as the fraction with the label A_D. Both metrics are computed per system across each dataset.

To compare the quality of HSA and DSA as measures of semantic accuracy, we manually annotated a sample of the data-text tuples. Since the vast majority of the text is expected to be accurate, especially on the cleaner datasets, we designed a sampling methodology to give a balanced representation of semantically accurate and inaccurate texts. To start, we sample 4 indices from the target dataset such that the human baseline outputs for these indices are labeled as: $\{(A_H, E_D), (E_H, A_D), (E_H, E_D), (A_H, A_D)\}$. We do the same with the state of art system and DATATUNER_FC outputs. We continue in a round-robin fashion until we get 24 indices per dataset. In the case of the LDC2017T10 dataset, we sample 24 indices in a similar fashion while ignoring the A_H and E_H labels. Next, two of the authors were presented with the input meaning representation and the output text generated by each system, for the 24 sampled dataset entries. The texts were shown in randomized order, similar to the Mturk study. The authors manually labeled each data-text tuple as accurate (A_M) or erroneous (E_M). The inter-annotator agreement measured with Cohen's Kappa was 0.81, indicating near-perfect agreement. We use these labels to assess the quality Q_D of the DSA metric as the percentage of cases where the manual label A_M matches A_D. Similarly, we evaluate the quality Q_H of the HSA metric as the percentage of cases where A_M matches A_H. These percentages are aggregated across systems, obtaining 120 samples per dataset. We present these metrics in Table 2.

DSA provides higher quality semantic annotations: We notice first that Q_D is 4.2% higher on ViGGO and 14.2% higher on both Cleaned E2E and WebNLG, compared to Q_H. These differences are statistically significant ($p < 0.05$) on WebNLG and Cleaned E2E as measured by McNemar's test (McNemar, 1947), where the null hypothesis is that the marginal probability for each outcome (*accurate* or *erroneous*) is the same for both algorithms. This provides more confidence in the ranking given by the DSA metric in Table 2 over the HSA one.

HSA struggles with open domains: The heuristic-based approach labeled only 41.2% of the human references in WebNLG as accurate, 16.9% lower than the score it assigned to our DATATUNER_FC. Since the latter was trained on human references, this difference is more likely to stem from the shortcoming of the heuristic-based approach in assessing the semantics. Checking the data, we observed that humans tend to create more diverse formulations, such as converting *United Kingdom* to *UK*, which are easy to miss with heuristics. On the contrary, our DSA metric scored the human references higher.

DATATUNER_FC delivers higher semantic accuracy: We also notice that, across all datasets, DATATUNER_FC significantly improves over the state of the art models as measured by the DSA metric (McNemar's (McNemar, 1947) gives $p < 0.05$). Compared to other DATATUNER variants, DATATUNER_FC also adds between 0.3% and 11.3% improvements, thus corroborating the utility of the semantic fidelity classifier.

8 Conclusion

In this work, we presented DATATUNER, an end-to-end data-to-text generation system equipped with an end-to-end semantic fidelity classifier. DATATUNER records new state of the art results on

four different datasets, with significant margins on automated metrics. We also show that our system has a clear fluency advantage over all the previous state of the art models. We further illustrate that DATATUNER provides strong accuracy on the task of delivering semantically consistent outputs.

References

Shubham Agarwal, Marc Dymetman, and Éric Gaussier. 2018. Char2char generation with reranking for the E2E NLG challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 451–456, Tilburg University, The Netherlands. Association for Computational Linguistics.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

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.

Thiago Castro Ferreira, Diego Moussallem, Emiel Krahmer, and Sander Wubben. 2018. Enriching the WebNLG corpus. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 171–176, Tilburg University, The Netherlands. Association for Computational Linguistics.

Andrew Chisholm, Will Radford, and Ben Hachey. 2017. Learning to generate one-sentence biographies from Wikidata. *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017 - Proceedings of Conference*, 1:633–642.

Edgar Dale and Jeanne S Chall. 1948. A formula for predicting readability: Instructions. *Educational research bulletin*, pages 37–54.

Marco Damonte and Shay B. Cohen. 2019. Structural neural encoders for AMR-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 3649–3658, Minneapolis, Minnesota. Association for Computational Linguistics.

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.

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.

Ondej 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*.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. 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.

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.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*, 7:297–312.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Juraj Juraska, Kevin K Bowden, and Marilyn Walker. 2019. Viggo: A video game corpus for data-to-text generation in open-domain conversation. In *Proceedings of the 12th International Conference on Natural Language Generation*.

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, New Orleans, Louisiana. Association for Computational Linguistics.

Kevin Knight, Laura Baranescu Bianca Badarau, Claire Bonial, Madalina Bardocz, Kira Griffitt, Ulf Hermjakob, Daniel Marcu, Martha Palmer, Tim O'Gorman, and Nathan Schneider. 2017. Abstract meaning representation (AMR) annotation release 2.0 LDC2017T10. *Web Download. Philadelphia: Linguistic Data Consortium*.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Evaluating the factual consistency of abstractive text summarization. *arXiv preprint arXiv:1910.12840*.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231. Association for Computational Linguistics.

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.

Percy Liang, Michael Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 91–99, Suntec, Singapore. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

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*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Amit Moryossef, Ido Dagan, and Yoav Goldberg. 2019a. Improving quality and efficiency in plan-based neural data-to-text generation. In *Proceedings of the 12th International Conference on Natural Language Generation*.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019b. 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, Minneapolis, Minnesota. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017a. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017b. The e2e dataset: New challenges for end-to-end generation. *arXiv preprint arXiv:1706.09254*.

Jekaterina Novikova, Ondrej Dušek, and Verena Rieser. 2018. RankME: Reliable human ratings for natural language generation. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 72–78, New Orleans, Louisiana.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Leonardo FR Ribeiro, Claire Gardent, and Iryna Gurevych. 2019. Enhancing amr-to-text generation with dual graph representations. 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)*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Anastasia Shimorina. 2018. Human vs automatic metrics: on the importance of correlation design. *arXiv preprint arXiv:1805.11474*.

Anastasia Shimorina and Claire Gardent. 2018. Handling rare items in data-to-text generation. *INLG 2018*, page 360.

Chris Van Der Lee, Albert Gatt, Emiel Van Miltenburg, Sander Wubben, and Emiel Krahmer. 2019. Best practices for the human evaluation of automatically generated text. In *Proceedings of the 12th International Conference on Natural Language Generation (INLG'19)*, Tokyo, Japan. Association for Computational Linguistics.

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 I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the*

IEEE conference on computer vision and pattern recognition, pages 4566–4575.

Frank Wilcoxon. 1992. Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pages 196–202. Springer.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019a. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019b. Transfertransfo: A transfer learning approach for neural network based conversational agents. *CoRR*, abs/1901.08149.

Jie Zhu, Junhui Li, Muhua Zhu, Longhua Qian, Min Zhang, and Guodong Zhou. 2019. Modeling graph structure in transformer for better AMR-to-text generation. 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 5458–5467, Hong Kong, China. Association for Computational Linguistics.

Appendix A: Model Diagram and Parameters

Figure 2 shows the full diagram of the D2T-LM model presented in Section 4.

In Table 3, we present examples of the outputs generated by DATATUNER and the state of the art models, alongside the human references.

D	Model	Examples
Cleaned E2E	Human	pub in riverside that also serves Italian food: The Vaults has got high prices, is not child-friendly and has an average rating. It is near Rainbow Vegetarian Caf in riverside.
	DATATUNER_FC	The Vaults is an Italian pub in the riverside area near Rainbow Vegetarian Caf. It has an average customer rating and a high price range. It is not child friendly.
	Dušek et al. (2019)	The Vaults is an italian pub with an average customer rating. It is located in the riverside area near Rainbow Vegetarian Caf. It is not children friendly and has a high price range.
WebNLG	Human	Adirondack regional airport serves both Lake Placid and Saranac Lake, New York. The length of the runway at Asirondack regional airport is 2003.
	DATATUNER_FC	Adirondack Regional Airport serves the cities of Lake Placid and Saranac Lake, New York and has a runway length of 2003.0.
	Castro Ferreira et al. (2019)	Adirondack Regional Airport serves the city of Lake Placid and Saranac Lake, New York and has a runway length of Shehbaz Sharif.
LDC2017T10	Human	the plan requires 8 precautionary steps before the order to shoot down a plane may be issued.
	DATATUNER_FC	the plan requires eight precautionary steps before the order to shoot down the plane can be issued.
	Zhu et al. (2019)	the plan required 8 precaution steps before it can be issued to order shot down.
ViGGO	Human	Guitar Hero: Smash Hits was a very bad game. 2009 was a terrible year for gaming and I just can't stand the games released that year.
	DATATUNER_FC	Guitar Hero: Smash Hits is a really bad game. 2009 was a really bad year for games
	Juraska et al. (2019)	Guitar Hero: Smash Hits is a very bad game, especially because it came out in 2009.

Table 3: Examples of text generated by the different models

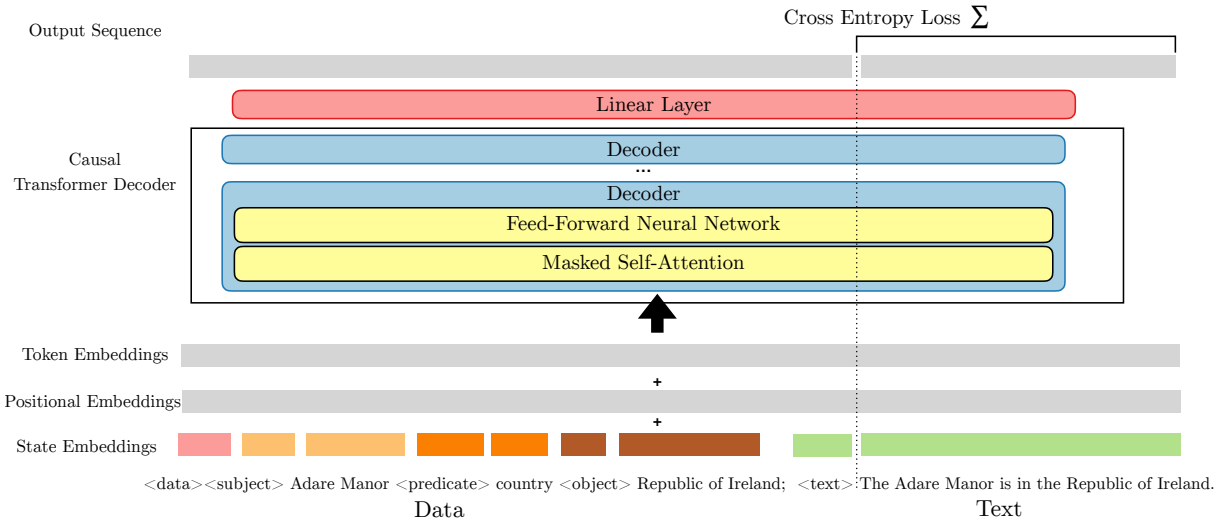

Figure 2: Data-to-text language model fine-tuning setup