Logic-Guided Message Generation from Raw Real-Time Sensor Data

Ernie Chang, Alisa Kovtunova, Stefan Borgwardt, Vera Demberg, Kathryn Chapman, Hui-Syuan Yeh

Department of Computer Science, Saarland Informatics Campus, Germany

Chair for Automata Theory, Technische Universität Dresden, Germany

{cychang@coli,s8kachap@teams,demberg@lst,yehhui@coli}.uni-saarland.de

firstname.lastname@tu-dresden.de

Abstract

Natural language generation in real-time settings with raw sensor data is a challenging task. We find that formulating the task as an end-to-end problem leads to two major challenges in content selection – the sensor data is both *redundant* and *diverse* across environments, thereby making it hard for the encoders to select and reason on the data. We here present a new corpus for a specific domain that instantiates these properties. It includes handover utterances that an assistant for a semi-autonomous drone uses to communicate with humans during the drone flight. The corpus consists of sensor data records and utterances in 8 different environments. As a structured intermediary representation between data records and text, we explore the use of description logic (DL). We also propose a neural generation model that can alert the human pilot of the system state and environment in preparation of the handover of control.

Keywords: message generation, content selection, domain variability, low resources, description logic, experiment

1. Introduction

Sensor technology has evolved in the last decade driven by the need to delegate routine tasks to machines. An example of such delegation is the Internet of Things (IoT) represented in wearables, smart homes, autonomous driving, etc. In the high-stakes applications, the system must continuously monitor the sensor data stream to detect any situation deterioration and react on it promptly. Moreover, in the case of a constantly changing environment, faithful data records can be very diverse and contain a lot of superfluous information. In this setting, it becomes challenging to detect an abnormality automatically.

In this work, we consider an example that embodies both of these aspects of sensor data: *redundancy* and *diversity*. Particularly, as a recent technological advance, drones with impressive features, advanced sensors and capabilities have become commonplace (Fuhrman et al., 2019) (*e.g.* for aerial surveys, mapping, aerial movies and even selfie-drones). The amount of sensor information routinely processed during a flight such as altitude, wind speed, air pressure, temperature, *etc.* is enormous. This is related to the fact that drones are extremely useful in the most remote and hard-to-reach places where very little can be controlled by human operators. As these drones are used for an increasingly wide range of tasks, interacting with drones becomes more important.

To enable these interactions, it is essential to devise a natural language generation (NLG) setup that can flexibly connect to a variety of data records collected by the drone and convey information reliably. In this paper, we propose a neural generation model (or drone assistant) that verbalizes messages from sensor data records in order to perform a controlled handover to a human drone pilot (see Figure 1). Recent data-driven methods have achieved good performance on various NLG tasks (Liu et al., 2018; Freitag and Roy, 2018; Chen et al., 2019). However, most studies focus on surface descriptions of simple record sequences, for example, attribute-value pairs of fixed or very limited schema, such as E2E (Novikova et al., 2017) and WikiBio (Lebret et al., 2016). In contrast, there is a much larger variety of data records available in the present setup, and the content selection task is substantially harder (only critical information, not all available information, should be mentioned at handover time).

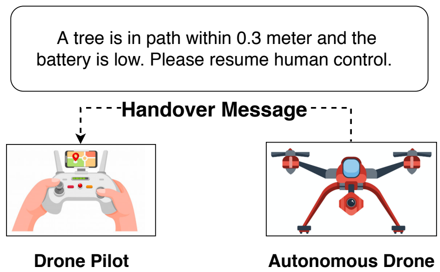

Figure 1. We focus on the drone handover as the main communicative function.

An on-device drone utterance generation model is thus faced with two challenges due to its *diverse* and *large* sensor data inputs (see Figure 2): (1) In real world scenarios, deployed drone dialogue systems are constantly exposed to drastically different environments; therefore, the ability of the system to generalize to diverse as well as unseen environments is desirable. (2) The redundancy of raw sensor records adds overheads to the encoder, and result in texts with low fidelity, where wrong facts are selected to be verbalized or even hallucinated.

To this end, we argue that it is necessary to leverage intermediate content representations to achieve faithful and controllable logical generation in such real-time settings with redundant data. In this paper, these repre-

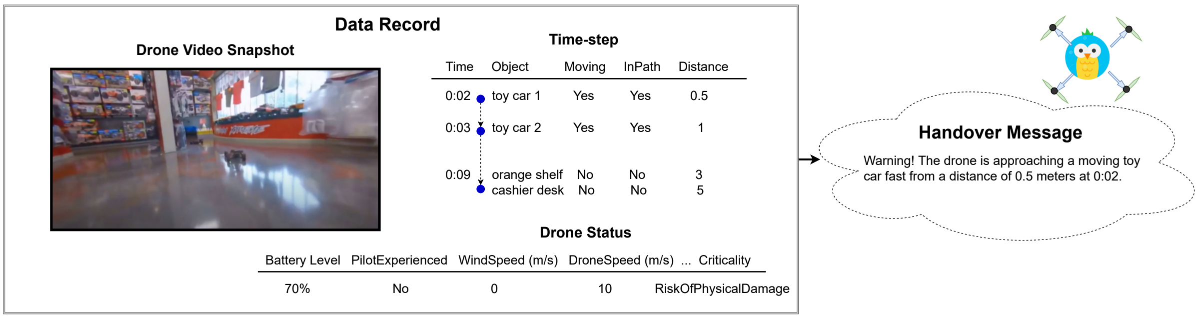

Figure 2. Depiction of one data sample from the dataset and the corresponding natural language utterance. For full drone status refer to Table 2.

sentations are generated using description logic (DL) ontologies (Baader et al., 2007). This *removes the burden of logical reasoning from the neural realization model* and allows for more flexible and high fidelity utterances to be produced. To allow us to study the utterance variability across different environments, we release a new dataset that consists of 316 data records derived from drone footage across 8 environments. The corpus offers insights into the challenges of real-time assistants using continuous streams of sensor data records.
In summary, our contributions are the following.

- We propose the first dataset that simulates real-world environments consisting of raw *drone sensor data records* paired with *DL annotations* and natural language *utterances*. We hope that our dataset encourages further research towards building real-time dialogue systems for large, real-time sensor data records.

- We develop a DL drone ontology and four queries to (i) automatically detect a critical situation and establish its urgency; (ii) if a handover is required, highlight justifiably relevant sensor records. These records combined into a DNF formula form a *DL expression*. A *DL annotation* combines the information acquired in (i)-(ii).

- For message generation, we use the DL annotations as intermediate content representation.

- We show the efficacy of our proposed technique for dealing with *diverse* and *redundant* raw sensor data. The code and the dataset are available online.[1]

2. Related Work

NLG from structured data or knowledge has been studied for many years. There are various applications, such as the automatic generations of weather reports (Liang et al., 2009), sport reports (Wiseman et al., 2017), or response generation in task-oriented dialogue systems (Wen et al., 2015; Budzianowski et al., 2018; Dušek et al., 2019).

Recent data-driven methods tend to conflate the pipeline modules into end-to-end neural networks, such as (Liu et al., 2018; Wiseman et al., 2017; Wiseman et al., 2018; Gong et al., 2019). However, purely neural models often suffer from problems with content fidelity (omission or halucination of facts) (Dušek et al., 2018). More recent work has begun to focus on preserving the fidelity of the generation, such as (Dhingra et al., 2019; Tian et al., 2019). Their work obtains good performance on surface-level NLG. In contrast, our work focuses on reducing content selection overheads for complex input data with high variability.

Recent NLG datasets mostly focus on surface-level generation. This includes WeatherGov (Liang et al., 2009), E2E (Novikova et al., 2017), WikiBio (Lebret et al., 2016), and ToTTo (Parikh et al., 2020). However, these datasets contain natural language sentences which are simple restatements of data records, and involve no abstract logical inference. In fact, the model in (Chen et al., 2020a) only obtains a 20% factual correctness rate based on human evaluation, which is far from an acceptable level in real-world systems. In contrast, our work focuses on the logical formulations executed on complex data records that can be derived from real-time systems realistically. To this end, we believe our new dataset can help future development of on-device real-time drone assistants.

3. Drone Sensor Data

This section describes the collected corpus and the simulated environments. We first describe the collection process in Section 3.1, then discuss the data schema in Section 3.2 and annotations (Section 3.4 and Section 3.5). In this work, the drone assistant is used in handover situations, where it sends a message to human pilots when there is a problem and the drone cannot continue flying autonomously. The type of handover is also categorized according to the *level of criticality*, which describes the drone's environment and corresponds to how urgent it is for control to be handed over to the human drone pilot.

[1] https://gitlab.com/erniecyc/drone/

3.1. Video Data Collection

We collected drone videos in 8 different environments: Disturbance (Di), Urban (Ur), Rural (Ru), Ocean (Oc), Desert (De), Island (Is), Factory (Fa) and Miscellaneous (Mi). These drone videos are recorded from the perspective of drones from either real drone manoeuvres or a drone simulator. The environments have drastically different settings; a detailed analysis is provided in Section 5. We split the original records into 316 snapshot videos of 10 seconds each. They are selected based on human judgement of whether the level of criticality rises to the point where a handover is required.

3.2. Data Record Schema

Each snapshot video from Section 3.1 is then manually annotated with realistic data records, which are based on the supposed sensor data that a drone can capture. We show an example of the data in Figure 2, which consists of a *time step* record of nearby objects, and a separate *drone status* record. The *time step* data reports 9 attributes that show the dynamics of the surrounding objects; for example, the object type, along with other information related to the flight path, such as **InPath** or **Moving**. The time step data are collected at 1-second intervals. The *drone status* record remains the same during the snapshot, as it indicates information of more permanence; for an example see Table 2. Together, they constitute a *snapshot* of data covering up to a 10-second interval. Snapshots are used as input data to the drone assistant.
In this section, examples of such data are written in **bold**.

3.3. Challenges

An end-to-end model using the raw snapshots as inputs faces the following problems.

1. The data record contains variable length information as the number and types of detected objects change between videos.

2. As the data is *permutation invariant* (Lee et al., 2019), the output of the model *should not change* under any permutation of the elements in the input data record.

3. Snapshots are long-form (containing at least 30 cells each). Irrelevant information in the data will tend to confuse the model.

4. By its design, transformer-based models are unable to process long sequences due to their self-attention operation, which scales quadratically with the sequence length.

To address these challenges, we incorporate DL reasoning in our drone assistant.

3.4. Annotation with Description Logic

Here, we describe the process of criticality annotation, where the *type* (see Table 1) and *level* (“informative”, “warning”or “advisory”) *of criticality* as well as DL expressions are added to each snapshot, in order to achieve more robust text generation.
Criticality prediction determines the type of utterance intent. To determine the type and the level of criticality, we employ *description logic* reasoning (Baader et al., 2007), based on an *ontology* consisting of *axioms* that describe background knowledge about drones and surrounding objects. For our test scenarios, the hand-crafted ontology[2] contains 62 predicates and 55 axioms. We use *ontology-mediated queries* that determine whether a certain critical situation is present in the input data (Borgida et al., 2003; Bienvenu and Ortiz, 2015). In the following text, ontology axioms and queries are in sans serif. For example, the ontology contains axioms $\mathsf{Foggy} \sqsubseteq \mathsf{LowVisibility}$ and $\exists \mathsf{env.LowVisibility} \sqcap \exists \mathsf{near.Object} \sqsubseteq \mathsf{RiskOfPhysicalDamage}$, which characterize fog as a visibility impairment and describe a critical situation of the drone flying close to another object in a low-visibility environment. The query predicate $\mathsf{RiskOfPhysicalDamage}$ indicates an increased criticality.
The general process of DL annotation works as follows. Based on the data record schema from Section 3.2, domain experts create a mapping from the records to the DL ontology predicates. Using the four query predicates from Table 1, for each snapshot DL reasoning can then automatically derive which *type of criticality* holds. For the most common criticality in the video collection, $\mathsf{RiskOfPhysicalDamage}$, we distinguish three *levels* of urgency depending on the ontology axiom triggering the criticality. We break ties between multiple reasons for criticality by keeping the most compelling one, *i.e.* between “informative” and “advisory” we choose the latter. Additionally, DL *justifications* (Horridge, 2011) are used to extract those parts of the input record that are responsible for the criticality. This information is encoded here into *DL expressions*, which takes the form of grounded DNF formulas (disjunctive normal form) expressing all reasons for the positive evaluation of the criticality queries. In the prototype implementation, since the ontology and the criticality queries are fixed, we did not use a DL reasoner to perform query answering. Instead we implemented the whole procedure as macros inside an annotation platform.

Example The partial status report in Table 2 is received from a defective drone steered by an inexperienced pilot inside a relatively cold room with different objects logged in Figure 2. Some of these data instances, on their own or in combination with others, indicate that safe piloting is not possible. The system must promptly recommend a handover. For

[2] https://cloud.perspicuous-computing.science/s/zLoBagLxo2fgqw4

Types of Criticality	Description	Example DL Expression
RiskOfPhysicalDamage	Potential physical damage (*e.g.* crash)	Altitude (m): 20 Battery_level: 30 OR InPath: true Distance: 3 at 00:02
RiskOfInternalDamage	Potential internal damage	weather: gloomy waterproof_drone: false
RiskOfHumanDamage	Risk of injuring nearby humans	indoor: true Distance: 0.5 Type: Human at 00:16
LostConnection	Drone connectivity/signal strength	Distance_from_remote_control (m): 162 Battery_level: 0

Table 1. Information on the types of criticality.

Wind_speed (m/s)	0
Drone_speed (m/s)	10
Pilot_experienced	FALSE
Altitude (m)	20
Temperature (celcius)	5
Distance_from_remote_control (m)	16
Battery_level	70
Low_visibility	FALSE
Normal_frame	FALSE
weather	sunny
upside_down	FALSE
good_motor_condition	TRUE
going_backwards	FALSE
indoor	TRUE
waterproof_drone	FALSE
flying_over	ground

Table 2. Sample of a status report that is part of a data record.

O	Risk of physical damage! There is a skyscraper in the flight path of the drone at a distance of 2m.
P	Risk of physical damage! The drone has a damaged frame and is flying indoors. There's a skyscraper in the drone path at a distance of 2m.
	There is a damaged frame and a dangerous floor in the drone's flight path at a distance of 2m.
	The drone has a damaged frame and is flying indoors. Risk of physical damage! There's a skyscraper on the flight path of the drone at a distance of 2m.

Table 3. Examples of original (O) text and its three **T5**-paraphrases (P).

example, the ontology has axioms such as $\mathsf{Flying} \sqcap \mathsf{Not\_Normal\_frame} \sqsubseteq \mathsf{RiskOfPhysicalDamage}$ ("flying a defective drone raises the risk of the drone being physically damaged") and $\exists \mathsf{reachable}.\exists \mathsf{inPath}.\top \sqsubseteq \mathsf{RiskOfPhysicalDamage}$ ("an object located at a reachable distance on a drone trajectory raises the risk of the drone being physically damaged"). These axioms applied to the data infer a critical state. Then, we automatically trace back which data records trigger this conclusion and combine them into a DL expression. In this example, it would look as follows:

$$[\mathbf{Altitude(m)} : 20 \text{ AND } \mathbf{Normal\_frame} : \mathsf{FALSE}] \\ \text{OR } [\mathbf{Object} : \mathsf{toy\ car\ 1} \text{ AND } \mathbf{InPath} : \mathsf{Yes} \text{ AND } \\ \mathbf{Distance} : 0.5 \text{ AT } \mathbf{Time} : 0{:}02].$$

For moving objects, we include the identifier and the time stamp. At this stage, according to the mapping, the abstract property of Flying is replaced by the raw information of $\mathbf{Altitude(m)} : 20$, which confirms that the drone is in the air.

In a real-world system, there are many benefits of using a formal ontology to encode background knowledge. Since it has its own format, it is independent of the platform or the programming language. This allows an ontology to be viewed, extended, and debugged by domain experts regardless of the end application. Moreover, in general, such ontologies can also be learned (semi-automatically) from other sources such as annotated data, text, and alignment with high-level ontologies (Lehmann and Völker, 2014). Existing ontology editing platforms, *e.g.* Protegé[3], also incorporate tools for visualisation, automatic analysis and reasoning, such as query answering.

3.5. Collection of Natural Utterances

As ground truth, we employed human experts to label each snapshot with an utterance that describes the situation detected by DL. As discussed in Section 3.4, the type and level of criticality already determine the character of the utterances. However, the example above demonstrates that the criticality can be created by various combined reasons. For instance, an "advisory" criticality type RiskOfPhysicalDamage is intended to alert the human pilot to make prompt decisions regarding the flight course. An "informative" RiskOfPhysicalDamage communicates a suboptimal internal state of the drone, such as a low power level, after which the human pilot can decide how to act on it. At this stage, the human experts are able to prioritise and aggregate the information *e.g.* an utterance could contain a solution recommendation or a partial situation report containing the data to be changed.

Paraphrase Augmentation. To enrich the variability of the texts, we use **T5** to generate paraphrases of the texts. For each utterance, we generate an additional three sentences by varying the beam size during decoding. By obtaining 10 sentences initially, linguistic experts were prompted to select the top 3 sentences based on their *fluency* and the perceived *textual similarity* with the original reference. We display some examples in Table 3.

We next describe the approach for automatically generating such utterances.

4. The Approach

The neural drone assistant primarily consists of two modules described in detail in Sections 4.1-4.2 and in Figure 3. The first one is a *data record linearizer* where table-formatted records are converted into a linear string

[3] `https://protege.stanford.edu/`

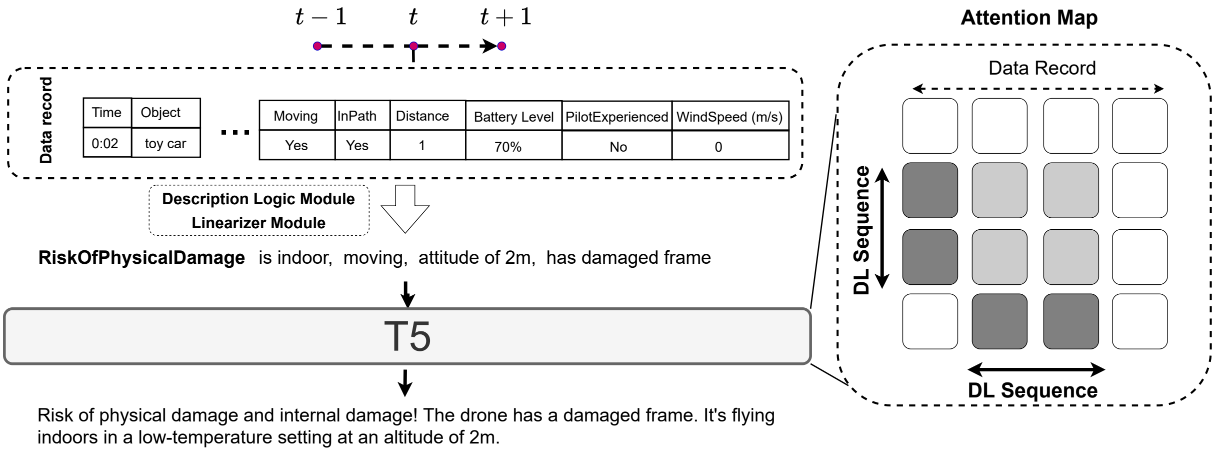

Figure 3. The **T5+DL** pipeline. A data record at time t is first processed by the DL module and linearized into the input sequence. This sequence is then fed into the **T5** model and used to generate an utterance. The attention map on the right demonstrates the DL-transformation of the data record being a *permutation invariant*, shorter subset sequence.

sequence along with auxiliary information. The second module, a *DL-to-text rewriter*, transforms the linearized sequence into a human-readable sentence. Finally, at the end of this section, we summarize the benefits of incorporating the DL reasoning seen from the NLG perspective.

Raw Data Records:
"frames": [{ "actions": [{ "act": "INFORM", "canonical_values": ["warning"], "slot": "risk_of_physical_damage", "values": ["warning"] }, { "act": "INFORM", "canonical_values": ["0.0"], "slot": "normal_frame", "values": ["0.0"] }, ...(**more rows omitted**)], "service": "Drone_1", "slots": [] }],

Linearized Data Records:
service_name=Drone_1,description={schema['description']},inform,name=risk_of_physical_damage, description=risk of physical damage or not, values = warning, inform, name=normal_frame, description=Normal frame or not, examples =1.0,0.0, values = 0.0, inform, ...(**more rows omitted**)

Figure 4. An abridged example of a data record *before* and *after* linearization.

4.1. Data Record Linearization

To linearize data records into sequences, we employ a technique previously used (Kale and Rastogi, 2020), where slot descriptions are added to each slot so as to ease the generation process (see Figure 4). While the slot descriptions are easy to obtain, it remains difficult to encode the semantics of large data records that contain irrelevant cells and duplicate information. Thus, we propose an extension of the schema-guided representation (Kale and Rastogi, 2020) by replacing the slot names with their natural language descriptions and also selecting them based on DL expressions (Section 3.4) so as to only focus on the relevant data records.

4.2. DL-to-Text Rewriting

The goal of the rewriting module is to convert DL expressions (generated as described in Section 3.4) to a natural language response with the same semantic content. Thus, we finetune a Text-to-Text Transfer Transformer (**T5**) (Raffel et al., 2019) model, which is a pretrained sequence-to-sequence transformer, to generate the natural language response using the linearized DL expression sequence as input. Figure 3 depicts the resulting framework. For ease of comparison, we perform 100-epoch updates for all training, as was empirically found to be sufficient for convergence.

4.3. DL Operation As Set Transformation

With the challenges from Section 3.3 in mind, we describe how the DL expression reduces the data complexity for **T5** in a way that is functionally similar to a **set transformer** (Lee et al., 2019).
T5 follows an encoder-decoder structure using stacked self-attention layers for both the encoder and decoder. Self-attention layers typically map one variable-length sequence of symbol representations $X = (x_1, \dots, x_n)$ to another sequence of equal length $(z_1, \dots, z_n)$, with $x_i, z_i \in \mathbb{R}^d$, for d being the embedding dimension of a word. The per-layer computational complexity of self-attention is $O(n^2 d)$[4] (Vaswani et al., 2017). By applying the *permutation invariant* data record linearization function based on DL expressions to the input data sequence, $\mathrm{DL}(X) = (x_{i_1}, \dots, x_{i_m})$, as a pre-processing step, we can decrease the length of layer input. Indeed, since $\mathrm{DL}(\cdot)$ is basically a filter, it guarantees that $m \leq n$ and, in practice (*e.g.* see the first two lines of Table 4), $m \ll n$. This results in much lower processing times

[4]In the transformer models, attention weights are calculated using all the words in the input sequence at once. Therefore, for estimating computational benefits of a new input size we can observe a difference already in the self-attention complexity.

and maintains a high level of representational power, thus yielding better *robustness* w.r.t. diverse input data. Similarly to how the approaches (Moryossef et al., 2019; Hua and Wang, 2019; Koncel-Kedziorski et al., 2019), the function $\mathrm{DL}(\cdot)$ performs an over-approximated text planning, which includes the selection of relevant content (*what to say*). Thus, *controllability*, measured by whether the generation correctly reflects the key semantic information in the input, improves naturally over the target output.

5. Corpus Analysis

This section and Table 4 present the details of our corpus. We begin by describing the high-level *characteristics* for each environment, then analyze both the *data record complexity* and *lexical richness* of the utterance.

Characteristics of Environments. Generally speaking, the generated types of criticalities as in Section 3.4 vary between environments due to differences in the types and numbers of objects, settings, distance between remote control and drone, etc. Those environments which contain more humans (*e.g.* Ur) present more human obstacles, which detect criticalities based on a different set of parameters than the non-human obstacles. The environments in more desolate areas (*e.g.* De, Oc) generally have fewer objects, thereby provoking fewer obstacle/nearby object warnings. Certain environments with open, outdoor settings (*e.g.* Ru, Is) contain more instances of long-distance remote drone control, and thus produce more LostConnection warnings than those in more enclosed environments (*e.g.* Fa, Mi). Lastly, some environments (*e.g.* Mi, Di) trigger substantially more RiskOfInternalDamage criticalities than others, which are typically prompted by a non-waterproof drone flying in a wet environment. Further, the two aforementioned environments are the only ones containing moving obstacles/nearby objects which warrant RiskOfPhysicalDamage warnings.

Data Record Complexity. Two crucial properties of the given data records are their *redundancy* and variable length when linearized. The performance of a neural drone assistant will very much be influenced by these factors since such records cannot be properly processed in a low resource setting. For instance, we observe that some attributes such as **going_backwards** are rarely used in the DL expressions or in the utterance: it should only add overhead to the encoding process in the rare occasion of the drone flying backwards. Thus, we compute the average number of cells per data record to get a sense of the distribution of raw data redundancy. We found that some environments (*e.g.* Di) tend to have more objects perceived by the drone, and so tend to have a larger time step record. Importantly, Table 4 indicates that in average the number of relevant cells for all environments is significantly reduced with the use of DL-transformation.

Lexical Richness. We used the Lexical Complexity Analyser (Lu, 2012) to measure various dimensions of lexical richness of the utterances from Section 3.5. We complement the traditional measure of lexical diversity type-token ratio (TTR) with the more robust measure of mean segmental TTR (MSTTR)[5] (Lu, 2012). The higher the value of MSTTR, the more diverse is the measured text. Table 4 shows that the highest MSTTR value is for Is and Mi while Di and Fa has the lowest value. In addition, we measure lexical sophistication (LS)[6], also known as lexical rareness and find that Oc has the highest LS score.

6. Experiments

We conduct experiments on the collected drone corpus that is split into training, validation and testing sets as reported in Table 5.

Vocabulary. We use SentencePiece (Kudo and Richardson, 2018) to encode text as WordPiece tokens. For all experiments, we use a vocabulary of 32,000 wordpieces as in **T5** (Raffel et al., 2019), which is shared across both the input and output of our model.

Configurations. Our baseline model **T5** is designed so that the encoder and decoder are each similar in size and configuration as in the previous work (Devlin et al., 2018). Specifically, both the encoder and decoder consist of 12 blocks, and each block comprises of self-attention, optional encoder-decoder attention, and a feed-forward network layer. The "key" and "value" matrices of all attention mechanisms have an inner dimensionality of 64 and all attention mechanisms have 12 heads, resulting in a model with about 220 million parameters. For regularization, we use a dropout probability of 0.1 everywhere dropout is applied in the model.

Test Scenarios. Based on the split in Table 5, we design four different test scenarios for the drone assistant model with different training and testing sets. For all inference scenarios, we either test the model on the data records of environments previously seen in the training set (**seen**) or not (**unseen**). In **all**, we train the model on data derived from all environments so that it obtains inductive bias that is more diverse and robust to environmental changes. We also simulate the scenario where we only have training data from one environment (**ind.**) or all except one. In the latter case, the model will be exposed to this **unseen** environment for testing. This is to simulate the real-time scenario where the drone assistant model is situated in new environments. For all scenarios, the drone assistant is to generate a handover message in relation to the input data record.

Benchmark Comparisons. In Table 6, we compare our model with Fairseq (Ott et al., 2019) **seq2seq** baseline, a simple **retrieval** method by using the data record and text pairs as a dictionary, and retrieving the text

[5] It divides the corpus into successive segments of a given length and then calculates the average TTR of all segments

[6] It is calculated as the proportion of lexical word types not on the list of 2,000 most frequent words from the British National Corpus.

	Di	Ur	Ru	Oc	De	Is	Fa	Mi
Average number of cells	168.85	74.90	39.46	27.92	12.56	27.6	34.0	52.25
Average number of cells (DL)	3.26	2.05	2.24	2.44	3.16	2.36	1.68	2.68
LS	0.60	0.58	0.59	0.63	0.60	0.61	0.62	0.62
MSTTR	0.56	0.60	0.58	0.59	0.60	0.61	0.56	0.61

Table 4. Corpus statistics across all environments. Lexical sophistication (LS) and mean segmental type-token ratio (MSTTR) are defined in Section 5. Average number of cells is defined as the average number of values per data record in the corpus.

	Di	Ur	Ru	Oc	De	Is	Fa	Mi
Train.	48	10	80	14	12	15	15	30
Valid.	6	5	10	5	5	5	5	5
Test.	6	5	10	5	5	5	5	5
Total	60	20	100	24	22	25	25	40

Table 5. Dataset splits in each environment.

of the closest[7] data record representation at inference time. A closely-related method is the template-based generator (**template**) where we construct variations of templates based on the training set. Lastly, we use **KGPT** (Chen et al., 2020b) which is a pretrained data-to-text model that learns to generate text from various types of structured data. We finetune this model on the training set.

We compare our approach, **T5+DL** described in Section 4, with a baseline **T5**. In the latter, the transformer model is simply finetuned on the linearized input sequence from each specified training set and tested on the target environments.

Linearized Data Record:
name=flying_over, description=Where the drone flying over, examples=ground,water,values = ground
name=risk_of_physical_damage, description=risk of physical damage or not, values = warning
name=risk_of_internal_damage, description=risk of internal damage or not, examples =1.0,0.0, values = 0.0
name=lost_connection, description=Lost connection or not, examples =1.0,0.0, values = 0.0 . . . (**more rows omitted**)

Linearized Data Record+DL:
name=risk_of_physical_damage,description=risk of physical damage or not, values = warning
name=object_inpath, description=Object is in path or not, examples =true, false, values = true
name=object_distance, description=Distance of object, values = 7

Reference: Risk of physical damage! The drone is flying with a damaged frame. Risk of physical damage! There's a tree in 7m in the drone's path.
T5+all : Distance of physical damage! The drone is flying toward a tree in its path 7m ahead.
T5+DL+all (Ours): of physical damage! The drone's frame is in need of repair. Risk of physical damage! The drone is flying toward a tree in its path 7m away.

Figure 5. An example of data records *before* and *after* DL-transformation. We display generation outputs of **T5+all** and **T5+DL+all** and show them side-by-side with the reference text.

7. Main Results

Here we present the experiment results and analysis by first (A) comparing different models within the same environments, so as to provide a more comprehensive comparison of benchmark systems and our proposed model. (B) We then examine how differences in environments influence the performance of each model. (C) Lastly, we also examine the impact of removing the testing environment from the training data, as a way to test the generalizability of the models. This is crucial to the development of real-time drone assistants as there should be no assumptions made about the type of environment that it will be exposed to.

(A) Benchmark Comparisons. Comparing our proposed approach with the benchmarks, we see that our proposed technique **T5+DL+all** outperforms all methods. In particular, **Retrieval+all** and **Template+all** achieve the worst performance; while **Seq2seq+all** and **KGPT+all** do slightly better. We find that our baseline approach without DL, **T5+all**, already generates utterances with more surface overlap with the reference than other techniques. However, the significant improvement is brought about with the inclusion of **DL** – with differences up to 37.36 BLEU points. This correlates with our other observation in Figure 5 where **Data Record+DL** is much shorter than the raw data records. It is also reflected in the poorer text quality of **T5+all**, which produces a shorter utterance and is missing some essential attributes. This shows that the combination of linearization technique, additional slot descriptions and DL transformation are highly beneficial for generating utterances with high reference surface overlap.

(B) Impact of Environments On Performance. Since the length of data records and vocabulary distribution is not homogeneous across different environments, it also results in differences in model performance. We observe that the *length of the data records influences the text quality very much*, as indicated by the improvements in terms of automatic evaluation *i.e.* BLEU-4 scores on **T5+DL+all** are generally higher than **T5+all**

[7]This is based on the string similarity.

Model	Di	Ur	Ru	Oc	De	Is	Fa	Mi
Template+all	22.43	41.87	37.62	34.27	26.58	22.11	34.82	38.63
Retrieval+all	18.91	45.20	40.80	31.89	24.62	21.89	33.34	42.77
Seq2seq+all	24.03	51.55	48.71	38.93	32.92	25.97	41.68	47.27
KGPT+all	25.14	54.50	50.28	40.01	34.52	28.22	45.51	50.93
T5+all	27.89	56.28	52.07	42.63	36.95	31.18	47.42	52.59
T5+DL+all	**65.25**	**65.88**	**78.61**	**52.45**	**47.15**	**59.65**	**63.30**	**71.89**
T5+ind.	15.88	33.90	40.47	28.16	33.04	22.06	**46.15**	35.56
T5+DL+ind.	**42.52**	**39.26**	**73.09**	**29.06**	**38.30**	**32.56**	42.61	**44.69**

Table 6. Performance in BLEU-4 (Papineni et al., 2002) on testing sets derived from the **seen** scenario across different environments. **Ind.** means that the training set is only drawn from the target environment. Scores of proposed approach are statistically-significant based on the two-tailed t-test with $p < 0.05$.

Model	seen			unseen		
	Nat	Miss	Wr	Nat	Miss	Wr
Human	4.76	0	0	4.58	0	0
Retrieval	3.32	49	63	2.95	57	48
Template	4.21	41	47	4.35	59	51
KGPT	3.45	46	66	4.23	48	44
Seq2seq	4.10	43	57	4.10	39	45
T5	4.20	45	51	4.15	49	51
T5+DL	**4.38**	**39**	**44**	**4.27**	**31**	**39**

Table 7. Human Evaluation on the sampled outputs (100 instances) for model comparison on **REF-B** for both **seen** and **unseen** scenarios across all environments. The first row is the (human) utterances from Section 3.5. We abbreviate Naturalness and Wrong as Nat and Wr.

across all environments. This is especially true after the DL-transformation, where the impact of length is seemingly "erased". For instance, almost all models perform relatively poor in Di at while the addition of DL boosted the BLEU score to 65.25.

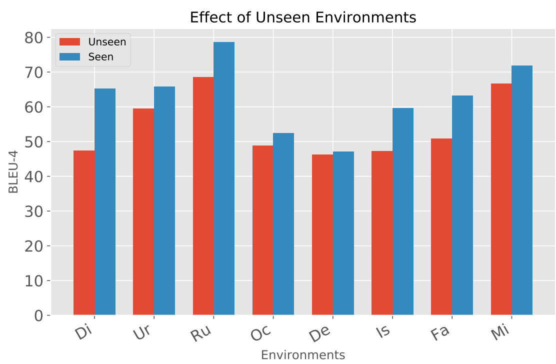

Figure 6. Bar chart to indicate the contrast between **seen** and **unseen** scenarios for our proposed approach **T5+DL+all**.

(C) On the Impact of Unseen Environments. For more realistic settings, we expose the trained model to environments unseen in Figure 6. The first observation is that the scores are generally lower for the same environment for the model and also across environments. This consistent degradation shows that unseen environments do require environmental-specific knowledge for target utterance generation. Since the degradation is less prominent on some environments, we attribute this to the difference in vocabulary distribution across environments, and to the intrinsic *robustness* that the model has with the use of the DL-transformation, where the long-form sequence is reduced to only its relevant subset, thereby alleviating the overhead of the encoding process.

Human Evaluation. We further ran a human evaluation on the model outputs (100 samples) to closely check the generation quality. Again, the six models are in consideration. Three annotators were asked to evaluate the outputs based on the *Naturalness* (0-5), *Miss* (the number of attributes in the data records that are missing in the outputs), and *Wrong* (the number of hallucinated attributes). We present the results in Table 7. We found that the scores are generally consistent with the automatic evaluation results (*i.e.* BLEU-4); our proposed approach outperforms the other ones by a large margin by generally yielding more natural utterances with fewer missing or wrong facts.

8. Conclusion

In this work, we present the task of message generation from real-time sensor data records and release a new language generation corpus that differs from previous corpora in terms of number and diversity in data records. Our results demonstrate the difficulty of the task such that it can serve as baseline for similar tasks where texts are generated from raw data records. Furthermore, we showed that description logic reasoning is able to transform sensor data records and reduce the difficulty of the encoding process to obtain better generation outputs.

9. Acknowledgements.

This work was supported by the DFG in grant 389792660 (TRR 248) (see `https://perspicuous-computing.science`).

10. Bibliographical References

Franz Baader, et al., editors. (2007). *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2 edition.

Bienvenu, M. and Ortiz, M. (2015). Ontology-mediated query answering with data-tractable description logics. In Wolfgang Faber et al., editors, *Reasoning Web. Web Logic Rules - 11th International Summer School 2015, Berlin, Germany, July 31 - August 4, 2015, Tutorial Lectures*, volume 9203 of *Lecture Notes in Computer Science*, pages 218–307. Springer.

Borgida, A., Lenzerini, M., and Rosati, R., (2003). *Description Logics for Databases*, page 462–484. Cambridge University Press, USA.

Budzianowski, P., Wen, T., Tseng, B., Casanueva, I., Ultes, S., Ramadan, O., and Gasic, M. (2018). Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In Ellen Riloff, et al., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 5016–5026. Association for Computational Linguistics.

Chen, Z., Eavani, H., Liu, Y., and Wang, W. Y. (2019). Few-shot NLG with pre-trained language model. *CoRR*, abs/1904.09521.

Chen, W., Chen, J., Su, Y., Chen, Z., and Wang, W. Y. (2020a). Logical natural language generation from open-domain tables. *arXiv preprint arXiv:2004.10404*, April.

Chen, W., Su, Y., Yan, X., and Wang, W. Y. (2020b). Kgpt: Knowledge-grounded pre-training for data-to-text generation. *arXiv preprint arXiv:2010.02307*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dhingra, B., Faruqui, M., Parikh, A. P., Chang, M., Das, D., and Cohen, W. W. (2019). Handling divergent reference texts when evaluating table-to-text generation. In Anna Korhonen, et al., editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4884–4895. Association for Computational Linguistics.

Dušek, O., Novikova, J., and Rieser, V. (2018). Findings of the e2e nlg challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 322–328.

Dušek, O., Novikova, J., and Rieser, V. (2019). Evaluating the state-of-the-art of end-to-end natural language generation: The E2E NLG Challenge. *arXiv preprint arXiv:1901.11528*.

Freitag, M. and Roy, S. (2018). Unsupervised natural language generation with denoising autoencoders. In Ellen Riloff, et al., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3922–3929. Association for Computational Linguistics.

Fuhrman, T., Schneider, D., Altenberg, F., Nguyen, T., Blasen, S., Constantin, S., and Waibe, A. (2019). An interactive indoor drone assistant. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6052–6057.

Gong, H., Feng, X., Qin, B., and Liu, T. (2019). Table-to-text generation with effective hierarchical encoder on three dimensions (row, column and time). In Kentaro Inui, et al., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3141–3150. Association for Computational Linguistics.

Horridge, M. (2011). *Justification Based Explanation in Ontologies*. Ph.D. thesis, University of Manchester, UK.

Hua, X. and Wang, L. (2019). Sentence-level content planning and style specification for neural 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 591–602, Hong Kong, China, November. Association for Computational Linguistics.

Kale, M. and Rastogi, A. (2020). 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.

Koncel-Kedziorski, R., Bekal, D., Luan, Y., Lapata, M., and Hajishirzi, H. (2019). Text Generation from Knowledge Graphs with Graph Transformers. 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 2284–2293, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Kudo, T. and Richardson, J. (2018). Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Lebret, R., Grangier, D., and Auli, M. (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, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1203–1213.

Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., and Teh, Y. W. (2019). Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, pages 3744–3753. PMLR.

Lehmann, J. and Völker, J. (2014). *Perspectives on Ontology Learning*, volume 18 of *Studies on the Semantic Web*. IOS Press.

Liang, P., Jordan, M. I., and Klein, D. (2009). Learning semantic correspondences with less supervision. In *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore*, pages 91–99.

Liu, T., Wang, K., Sha, L., Chang, B., and Sui, Z. (2018). Table-to-text generation by structure-aware seq2seq learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 4881–4888.

Lu, X. (2012). The relationship of lexical richness to the quality of esl learners' oral narratives. *The Modern Language Journal*, 96(2):190–208.

Moryossef, A., Goldberg, Y., and Dagan, I. (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, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Novikova, J., Dusek, O., and Rieser, V. (2017). The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, Saarbrücken, Germany, August 15-17, 2017*, pages 201–206.

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. (2019). fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (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.

Parikh, A. P., Wang, X., Gehrmann, S., Faruqui, M., Dhingra, B., Yang, D., and Das, D. (2020). Totto: A controlled table-to-text generation dataset. *CoRR*, abs/2004.14373.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Tian, R., Narayan, S., Sellam, T., and Parikh, A. P. (2019). Sticking to the facts: Confident decoding for faithful data-to-text generation. *CoRR*, abs/1910.08684.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *NeurIPS*, pages 5998–6008.

Wen, T., Gasic, M., Mrksic, N., Su, P., Vandyke, D., and Young, S. J. (2015). Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In Lluís Màrquez, et al., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1711–1721. The Association for Computational Linguistics.

Wiseman, S., Shieber, S. M., and Rush, A. M. (2017). Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2253–2263.

Wiseman, S., Shieber, S. M., and Rush, A. M. (2018). Learning neural templates for text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3174–3187.