xR4DRAMA

Extended Reality For DisasteR management And Media planning

H2020-952133

D3.6

Multilingual information generation techniques

Dissemination level:	Public
Contractual date of delivery:	Month 13, November 31, 2021
Actual date of delivery:	Month 14, December 1, 2021
Work package:	WP3 Analysis and fusion of multi-modal data
Task:	T3.6 Personalized information generation
Type:	Demonstrator
Approval Status:	Final version
Version:	1.0
Number of pages:	36
Filename:	D3.6_xR4Drama_MultilingualInformation GenerationTechniques_v1.0.pdf

Abstract

Deliverable 3.6 describes the progress on the task T3.6 “Personalized information generation” of WP3. This task accounts for the production of multilingual text generation in the xR4DRAMA platform. The component involved in this task is the report/text generation module, which receives input from the multimodal information fusion (T3.5) to be realised as natural language sentences in three languages (English, Italian, German). The advances during

the first half of the project are discussed in the course of the deliverable, namely: (i) the definition of the initial relevant information to be supported by the xR4DRAMA platform on the generation side, (ii) the compilation and extension of the linguistic resources for natural language generation (lexica, graph transduction grammars, datasets) for English, and (iii) some experiments with statistical generation and new datasets and methods for the evaluation of automatically generated text.

co-funded by the European Union

History

Version	Date	Reason	Revised by
0.1	29/09/2021	Table of contents	Simon Mille
0.2	29/11/2021	Draft for internal review by Maria Pacelli (STX)	Simon Mille
1.0	01/12/2021	Final version	Simon Mille

Author list

Organization	Name	Contact Information
UPF	Simon Mille	simon.mille@upf.edu

Executive Summary

Deliverable 3.6 describes the progress on the task T3.6 “Personalized information generation” of WP3. This task accounts for the production of multilingual text generation in the xR4DRAMA platform.

The component involved in this task is the report/text generation module, which receives input from the multimodal information fusion (T3.5) to be realised as natural language sentences in three languages (English, Italian, German). The advances during the first half of the project are discussed in the course of the deliverable, namely: (i) the definition of the initial relevant information to be supported by the xR4DRAMA platform on the generation side, (ii) the compilation and extension of the linguistic resources for natural language generation (lexica, graph transduction grammars, datasets) for English, and (iii) some experiments with statistical generation and new datasets and methods for the evaluation of automatically generated text.

Abbreviations and Acronyms

DSyntS	Deep Syntactic Structure
FORGe	Fabra Open-source Rule-based Generator
JSON	JavaScript Object Notation
KB	Knowledge Base
MorphS	Morphological Structure
NLG	Natural Language Generation
P1	First prototype
PoS	Part of Speech
PredArg	Predicate-Argument
SemS	Semantic Structure
SSyntS	Surface Syntactic Structure
UR	User Requirement
VR	Virtual Reality

Table of Contents

1 INTRODUCTION

This deliverable presents the work done for Natural Language Generation (NLG) since the inception of the project. NLG corresponds to Task 3.6, which officially started on M7, and aims at the development of a multilingual report generator. UPF's generator takes as input the fused contents of the Knowledge Base (KB, T3.5), and targets the production of texts in English, Italian and German.

According to the methodology described in the DoA, there are three main tasks related to NLG: (i) content selection, (ii) discourse planning, and (iii) linguistic generation. The role of content selection is to decide which parts of the KB need to be verbalised; this task will be mostly addressed in the second part of the project in accordance with user needs; in this deliverable, we assume that all the information in the KB is selected for producing the reports. The role of discourse planning is to organise discursively the contents of the inputs to be generated, in particular to establish discourse relations between utterances to be produced by the agent. It is referred to as "semantic aggregation" or "sentence/information packaging" in this deliverable. Linguistic generation is addressed (i) by building on the UPF grammar-based generator FORGe (Mille et al. 2019) to allow for multilingual generation of more or less complex structured data using a wide range of linguistic resources, and (ii) by developing statistical generation modules by combining state-of-the-art machine learning techniques with the rule-based generator or substitute some subcomponents of the rule-based generator to increase its efficiency. During the first half of the project, we mostly report on grammar-based generation in English, although a large part of the rule engine is multilingual. The developed generators will be both evaluated on international benchmark datasets and on xR4DRAMA data in the next deliverable.

In this document, we present a brief state of the art (Section 2), a study of the contents to be verbalised in xR4DRAMA and how they are rendered (Section 3), a description of the approach followed for generating the texts (Section 4) and a summary of the improvements to the tools, datasets and methods achieved during the first year (Section5).

2 STATE OF THE ART

For targeting the development of a reusable Natural Language Generation (NLG, also referred to as Text Generation) pipeline and its interface with the Knowledge Base (KB, see WP5), we base our approach on the traditional view of NLG as a sequence of three subtasks: (i) content selection, which is responsible for determining the contents to be rendered as text, (ii) text planning, which takes care of packaging the contents into discursively organised units (i.e., sentences), and (iii) linguistic generation, which realises the contents as well-formed text (Rambow and Korelsky 1992). In our approach, another module is added between the first two in order to account for mapping the language-agnostic Knowledge Graphs onto linguistic structures, and linguistic generation is in its turn split into several modules that address separately sentence structuring (choosing the words to be used and organise them syntactically), word ordering and morphological agreement resolution. In xR4DRAMA, step (i) will be carried out by a dedicated module, and steps (ii) and (iii) by the text generation module described below.

In general, each step can be performed using template-based, grammar-based or statistical systems, or a combination of these (Ballesteros et al., 2015, Gardent et al. 2017). Currently, a lot of research on the topic addresses the whole sequence as one step, and focuses on filling the slot values of pre-existing templates using neural network techniques (Nayak et al. 2017). Few systems follow a theoretical framework, and most of them make extensive use of language models (i.e., use a large amount of reference texts) to statistically mimic correct language use (Castro Ferreira et al. 2020, Chen et al. 2020, Zhao et al. 2020) The main problems with these approaches are, as mentioned above, their low portability to new languages and domains and the lack of control over the output, but also their sensitivity to biases and the very limited amount of linguistic knowledge used during the generation process. A grammar-based generator does not require training material, allows for a greater control over the outputs (e.g., for mitigating possible errors or tuning the output to a desired style), and the linguistic knowledge used for one domain or language can be reused for other domains and languages. However, due to their complexity, such approaches have undergone few developments within the open-source community in the recent years (Gatt and Krahmer, 2018) and most are based on the SimpleNLG framework (Gatt and Reiter 2009), which requires a high degree of tailoring of the input (Moussalem et al. 2018). The only grammar-based system used successfully in recent NLG shared tasks is FORGe (Mille et al. 2019), an open-source generator developed by UPF which is implemented as graph-transducers in the MATE platform (Bohnet and Wanner, 2010) and covers the last two NLG subtasks (text planning and linguistic generation). It was the best system at the WebNLG 2017 task (automatic verbalisation in English of 400 DBpedia properties) according to all human evaluations, and was the most portable generator, with the best results for all metrics on unseen data, that is, on inputs for which no training data had been provided. Although promising, FORGe's coverage is limited to the use cases of the projects it has been developed on, its text planning layer is embryonic, and despite achieving very high accuracy and grammaticality, the fluency of the generated texts can be improved, as shown in a recent large-scale evaluation (Castro Ferreira et al. 2020). Furthermore, it has been only partially adapted to German and Italian. It is one of the objectives of the project to significantly improve the generator in terms of coverage and quality of generated texts.

The generation as we envision it starts from the conceptual structures, and consists of a sequence of graph-transduction grammars that map successively the conceptual, or Predicate-Argument (PredArg) templates to linguistic structures of different levels of abstraction, in particular syntax, topology, morphology, and finally surface texts. PredArg structures are very similar to the Facts in ILEX's Content potential structures (O'Donnell et al., 2001), or the Message triples in NaturalOWL (Androutsopoulos et al., 2013), with the difference that all predicates in the PredArg structures are intended to represent atomic meanings, allowing for more flexible aggregation and sentence structuring. The first part of the generation pipeline, which produces aggregated predicate-argument graphs, is also comparable to ILEX, while the surface realisation is largely inspired by MARQUIS (Wanner et al., 2010). Our generator shares not only its general architecture with these two systems, but also the use of lexical resources with subcategorisation information and of a multilingual core of rules. One of the specificities of our pipeline is that two types of aggregation take place during generation, one at the predicate-argument level (in a NaturalOWL fashion), and one at the syntactic level (see Section 4.2 for more details).

As far as input representations are concerned, an NLG pipeline needs to be fed with linguistic structures. These are quite different from the triples found in the KB, in which the properties are labelled with an open vocabulary and only two types of relations (Subject and Object) are used. The triples must be mapped onto linguistic concepts and relations, preferably according to standard lexico-semantic resources such as VerbNet (Schuler 2005), NomBank (Meyers, et al. 2004), or PropBank (Kingsbury and Palmer 2002) to ensure reusability. These resources can be used as interlingua thanks to the amount of multilingual resources connected to them. To the best of our knowledge, little research has been carried out so far on bringing together KB contents and standard linguistic resources in the context of NLG: on the one hand, standard Semantic Web approaches such as Lemon (Walter, Unger and Cimiano 2014) or word-embeddings-based lexicalization (Perez-Beltrachini and Gardent 2016) define their own lexicons to be associated with the properties, and on the other hand, linguistic resources such as VerbNet, NomBank and PropBank are not connected with reusable Knowledge Bases. Finally, even if the Semantic Web components were mapped to NomBank and PropBank entries, the syntactic information about the participants is not expressed in these resources. This subcategorisation information can be derived from VerbNet, which is neither NLG nor dependency-oriented, so the task is challenging.

3 INITIAL STUDY OF CONTENTS TO GENERATE

In this section, we present the user requirements as defined in D6.2, and how they are individually mapped to natural language utterances (Section 3.1); we then briefly describe additional data available in xR4DRAMA's Knowledge Base (Section 3.2).

3.1 User Requirements

In this section, we present all the User Requirements (URs, as found in D6.2) that can be presented to the users under the form of text. Note that (i) not all of them are currently addressed, since a specific data source may not be available at this point in the project (e.g., satellite and disaster image analysis), and (ii) in the future, not all of them may be addressed, since users may prefer another output format (table, image, etc.) or pre-generated texts (e.g., mitigation information). In Tables 1 to 7 below, each User Requirement as found in D6.2 is considered as a data point. For each requirement/data point we provide a sample verbalisation that will be used for xR4DRAMA's first prototype (P1, see next sections). In grey, the URs that are not addressed at this point, thus not covered by the language generation module for P1. The colours in the captions match the colours in the Section 4 for easier readability.

Table 1: URs related to **Accessibility** and their verbalisation

Info-ID (D6.2)	Data point description (D6.2)	Sample P1 standalone verbalisation
G-01	quality and type of road (highway, street, path), distance to railway station and airport, public transport	TBD
PUC1-14	Possibility to define an appropriate escape route or a suitable way to reach an intervention are	TBD
PUC2-03	availability of parking	There is a parking on the site.

Table 2: URs related to **Geography/Surroundings** and their verbalisation

Info-ID (D6.2)	Data point description (D6.2)	Sample P1 standalone verbalisation
G-02	the shape, look and size of buildings, the purpose of buildings	A structure has been detected.
		The structure is a tower.
G-03	indication of high voltage lines, windmills and other landmarks	There is a windmill in the surroundings.
G-04	indication of roads, highways, railroads	There is a highway in the surroundings.
PUC1-01	indication of rivers, water courses, riverbanks	There is a river in the surroundings.
PUC1-02	indication of manholes, electrical and gas pipes	There is an electrical pipe in the surroundings.
PUC1-015	Information derived by satellite images analysis	TBD

Table 3: URs related to Environmental factors and their verbalisation

Info-ID (D6.2)	Data point description (D6.2)	Sample P1 standalone verbalisation
G-05	basic weather information through the year or a specific period of time	Dropped by consortium
PUC1-10	Information on environmental variables: water level, rain, temperature, humidity	TBD
PUC1-11	Information available on radar meteo	TBD
PUC2-01	identification of possible sources like busy roads or highways, crowds of people, factories, airports, railway stations, railway tracks	There is a source of noise pollution (highway) on the site.
PUC2-02	identification of possible sources like streetlights, ads etc.	There is a source of light pollution (street light) on the site.
PUC2-07	simulation of the course of the sun during a day	TBD
PUC2-14	the noise situation on site recorded by the location scout via a Smartex device as mp3-file	TBD

Table 4: URs related to General information and their verbalisation

Info-ID (D6.2)	Data point description (D6.2)	Sample P1 standalone verbalisation
PUC1-03	Information on the presence of areas of attention, safe waiting/parking places, shelters, sand-bag distribution areas	There is a parking on the site. **NO variant:** There are no shelters on the site.
PUC2-06	textual information on specific sites/buildings in the area of interest	TBD
PUC2-13	Information on the security situation in the designated country	TBD

Table 5: URs related to Flood risk management and their verbalisation

Info-ID (D6.2)	Data point description (D6.2)	Sample P1 standalone verbalisation
PUC1-04	Raster data of flow velocity and water depth in flood scenarios	TBD
PUC1-05	Information on flood risk level in the territory	A flood has been reported in the city center (high risk level).
PUC1-06	Information about flood reports localised by audio analysis and categorised according to the problem issue	TBD
PUC1-07	Information on flooded elements (e.g. cars and people inside the river)	6 persons are in danger.
PUC1-08	Information related river embankments overtopping or breaking	TBD

PUC1-09	Information on the presence of elements at risk and the degree of emergency	*See G-02 above.*
PUC1-16	Information on the potential presence of people in areas at risk	*See PUC-1-07 above.*
PUC1-17	Information on the potential presence of cultural heritage/natural sites	*See G-02 above.*
PUC1-18	Information on the localisation, type of action, activation threshold of the Vicenza Risk Management Plan procedures	TBD

Table 6: URs related to **Human factors** and **Legal issues** and their verbalisation

Info-ID (D6.2)	Data point description (D6.2)	Sample P1 standalone verbalisation
PUC1-12	Physiological parameters of first responders in the field (e.g. activity, EEG, respiration)	A first responder is running.
PUC1-13	detect by stress analysis the stress level in first responders affected by flooding/involved in rescue operations	A low stress level has been detected.
		The confidence of the stress level is 65%.
PUC2-04	necessity of a permission for filming on the ground with a crew	TBD
PUC2-05	type of permission for filming with drones, possible restrictions for filming	TBD

Table 7: URs related to **Facilities** and **Simulation** and their verbalisation

Info-ID (D6.2)	Data point description (D6.2)	Sample P1 standalone verbalisation
PUC2-08	availability and accessibility of power outlets	There are (no) power outlets on the site.
PUC2-09	availability and accessibility of bathrooms	There are (no) bathrooms on the site.
PUC2-10	list of/indication of available places to eat/drink	There are (no) restaurants on the site.
PUC2-11	Possibility to put props/decoration/etc. in the environment	Installing decorations is possible on the site.
PUC2-12	Possibility to simulate various flights of drones in VR	Flying drones is possible on the site.

3.2 Additional contents modelled in the Knowledge Base

The current version of the Knowledge Base (KB, as of October 2021) covers some information related to Geography/Surroundings (Table 2), Flood risk management (Table 5) and Human

Factors (Table 6).[1] It also contains the raw data from sensor measurements, as shown in Table 8, and confidence scores (probabilities) associated to, e.g., visual analysis and stress detection outputs. It is unclear at this point to what extent this data will need to be verbalised.

Table 8: General representation of **raw sensor data**

KB data classes	Value type
Observation time (sensor data)	dateTime
Observation ID (sensor data)	string
Observation result (sensor data)	integer

[1] In other words, not all contents described in this deliverable may be visible in xR4DRAMA's first prototype reports.

4 BASIC TECHNIQUES FOR NATURAL LANGUAGE GENERATION

In this section, we present an input/output pair of the natural language generation module (Section 4.1)

4.1 Input to the generation module

The structured information described in Section 3 needs to be mapped to written reports to be presented to the users. A user can require a report over the whole KB, or just updates with respect to the previous report, or be notified that new data is available from one of the sources (text, images, sensors, etc.). The relevant contents from the KB (and only them) are thus selected accordingly, and stored in a structured JSON file which is then sent to the language generation module. In Figure 1 below, we provide an example in the JSON format of an input that includes all the data points currently covered by the language generation module, with simulated values for each data point. The colors of each data point match the colors of the data points in Tables 1-7 above (Section 3.1).


```
{
  "header": {},
  "body": {
    "riskLevel": {
      "risk": ["flood"],
      "riskLevel": ["high"],
      "location": ["Zone 2"]
    },
    "responderAction": {
      "action": ["stand", "run"],
      "numberResponders": ["1", "7"]
    },
    "stressLevel": {
      "stressLevel": ["high"]
    },
    "stressLevelConfidence": {
      "stressLevelConfidence": ["95"]
    },
    "vehicleInDanger": {
      "vehicleInDanger": ["9"]
    },
    "peopleInDanger": {
      "peopleInDanger": ["26"]
    },
    "animalInDanger": {
      "animalInDanger": ["2"]
    },
    "buildingDetected": {
      "building": ["structure"]
    },
    "buildingType": {
      "building": ["structure"],
```

```
        "buildingType": ["bridge"]
      },
      "availabilityParking": {
        "availabilityParking": ["yes"]
      },
      "availabilityPower": {
        "availabilityPower": ["yes"]
      },
      "availabilityBathroom": {
        "availabilityBathroom": ["yes"]
      },
      "noisePollution": {
        "noiseSource": ["highway", "airport", "factory"]
      },
      "lightPollution": {
        "lightSource": ["antenna"]
      },
      "availabilityAttentionNo": {
        "areasOfAttention": ["large_shelter"]
      },
      "availabilityProps": {
        "availabilityProps": ["yes"]
      },
      "availabilityDrone": {
        "availabilityDrone": ["yes"]
      },
      "surroundingsLandmark": {
        "landmark": ["tower", "castle"]
      },
      "surroundingsRoad": {
        "road": ["road"]
      }
    }
  }
```


Figure 1: A sample input obtained from the KB representation

Verbalising each data point in isolation would lead to an accumulation of short and possibly repetitive statements as the ones found in the rightmost columns of Tables 1-7. The objective of the language generation module is to present the information in a more user-friendly way, by packaging the information into coherent sentences and a coherent text. A possible output of the language generation module for the input above is the following (following again the color coding of the data points):

A flood has been reported (high risk level). 1 first responder is standing and 7 first responders are running. A high stress level has been detected (95% confidence). 9 vehicles, 26 persons and 2 animals are in danger. A structure, which is a bridge, has been detected. There is a parking, power outlets and bathrooms but no large shelters on the site. There are sources of noise pollution (highway, airport and factory) and a source of light pollution (antenna) there.

Installing decorations and flying drones is possible on the site. There is a tower, a castle and a road in the surroundings.

In the next subsection, we detail the process of the verbalisation of the input structured data.

4.2 Grammar-based generation using the UPF-FORGe system

This section focuses on the extension of UPF's multilingual discourse generators, developed for multilingual report generation in a series of European projects, to an incremental report generator that is coordinated with xR4DRAMA's Knowledge Base. The work has been carried out as foreseen during the first 12 months, thus, the present deliverable contains a description of the approach (Section 4.2.1) and the respective implementation (Section 4.2.2).

4.2.1 Approach

In our approach, the text generation consists of two sub-modules: sentence packaging (aka text planning) and linguistic generation. The latter is split into several modules that address the tasks of sentence structuring (choosing the words to be used and organising them syntactically), word ordering, and morphological agreement resolution. The advantage of splitting text generation into specific tasks is to allow for a precise and independent modelling of each level of language description (semantics, syntax, topology, morphology). This is one of the central ideas of the Meaning-Text Theory (Mel'čuk 1988), which serves as a theoretical framework for the generator.

Text generation starts from the ontological assertions that comprise the selected contents of the Knowledge Base, and thus the ontological structures must be mapped to linguistic structures before the process starts. The generation is performed step by step, by successively mapping one level of representation onto the adjacent one:

Ontology (KB)

Conceptual Structure

Semantic Structure

Packaged semantic structure

Deep-Syntactic Structure

Surface-Syntactic Structure

Morphologic Structure

Sentence

In the following, we describe the role of each transition.

Projection of ontology constructs onto conceptual structures

When mapping to the conceptual structure, participating elements are mapped to linguistic predicates or arguments (i.e., nodes and/or labelled edges that link the predicate to the argument). For instance, in Figures 3 and 4 below, the predicates 'report' and 'risk_level' are introduced to verbalise the data point RiskLevel. 'Report' typically has 3 arguments: who reports (first argument, A1), what is reported (second argument, A2), and to whom it is

reported (third argument, A3); only the second argument conveys relevant information for xR4DRAMA, thus only the second argument is used. Circumstantials (e.g., temporal attributes of an action, such as start time and duration) are treated as non-predicative elements with a specific edge label “Location” or “Time” that links them to the element that the location or time is specified; for instance, in Figure 4, ‘Zone 2’ or ‘site’ are locations of where the flood was respected and where the facilities are available or not. The “NonCore” relation is used for non-argumental relations that are neither Location nor Time, and the ‘Elaboration’ relation is a discursive relation that indicates the presence of additional information on a given topic. As already mentioned, as opposed to the KB structure, the conceptual structure encapsulates the first version of what will be found in the final sentence: only the elements which will be mentioned (explicitly or not) are kept.

At this point in the project, the conceptual structures are in the form of simple predicate-argument templates associated with each data point in the KB. We crafted 23 predicate/argument (PredArg) templates to cover the data points described in Section 3. The variable slots in the templates (between square brackets) are filled/populated with the values of the input JSON. As an example, consider in Figure 2 a fragment of Figure 1 above and in Figure 3 the 3 PredArg templates that cover the 5 data points.


```
{
  "header": {},
  "body": {
    "riskLevel": {
      "risk": ["flood"],
      "riskLevel": ["high"],
      "location": ["Zone 2"]
    },
    "availabilityParking": {
      "availabilityParking": ["yes"]
    },
    "availabilityPower": {
      "availabilityPower": ["yes"]
    },
    "availabilityBathroom": {
      "availabilityBathroom": ["yes"]
    },
    "availabilityAttentionNo": {
      "areasOfAttention": ["large_shelter"]
    }
  }
}
```


Figure 2: A sample input obtained from the KB representation (fragment of Figure 1)

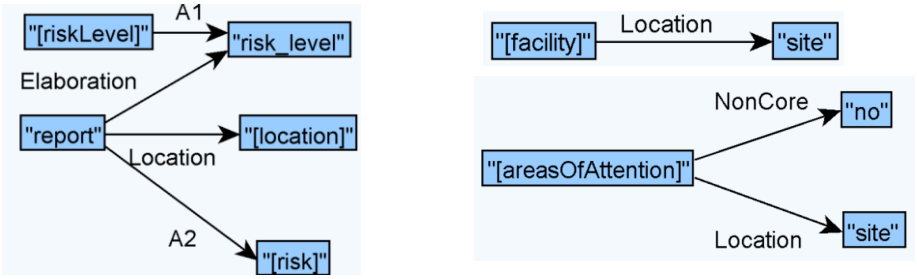

Figure 3: Three PredArg templates used for the 5 data points of Figure 2 (graphical representation of JSON-formatted templates)

The template in the top right of Figure 3 is used for the data points 'availabilityParking', 'availabilityPower' and 'availabilityBathroom'. Figure 4 shows the conceptual representation that corresponds to Figure 2; it consists of 5 populated PredArg templates.

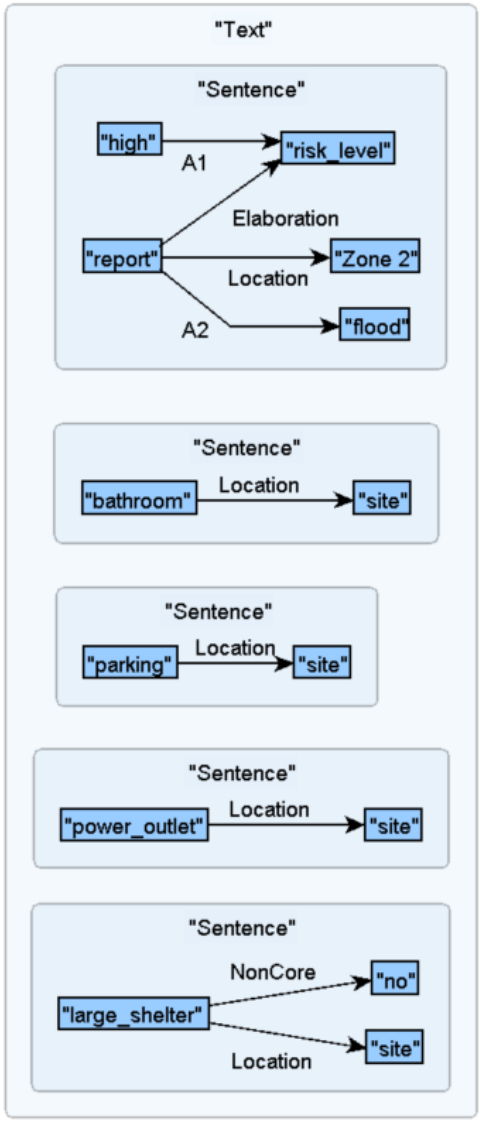

Figure 4: Sample conceptual representation

From Conceptual Structure to Semantic Structure (SemS): choosing the meanings in each language

The conceptual structure is mapped to a language-specific structure according to the available meanings (semantemes) in the concerned language (namely, English, Italian, depending on the needs of the respective use cases). In theory, a semanteme can be lexicalised by many different words, see for instance the semantic dictionary entry 'CAUSE': CAUSE { lex = cause_N | lex = cause_V | lex = contribute |lex = responsible | lex = due | lex = because | etc.}. However, in xR4DRAMA, a simplified and more practical view has been applied, considering the lexical units (i.e., words, as opposed to meaning units) such as 'cause_V' (cause as a verb) are the basic meaning units in the semantic structure. In practice, most of the time the semantic structure simply serves for the introduction of the lexical units in the target language, so the semantic structure for the generation in English in our running example would be the same as in Figure 4.

The semantic structure is unambiguous: each semanteme is the argument of a predicate and is numbered by the valency (or subcategorisation frame) of the predicate, through the relation linking the two of them. Each language has its own set of predicates, and each predicate has its own valency.

Text planning / Sentence packaging: defining the boundaries of sentences

If several discursive units as shown in Figure 4 (the group of nodes called "Sentence" is what we call here a discursive unit) are provided to the generator, each of them will be realised by default as an independent sentence. In order to group different units into complex sentences, we need to perform an "aggregation", or a "packaging" of the information, in two steps. First, we look for shared pairs of predicate and subject/object argument or location in the input: if elements of two unlinked predicates have the same relation with their respective predicates, they will be coordinated. For instance, four of the discursive units share the same location ('site'); the first three have the same node configuration and are thus aggregated together (introduction of a coordination with 'and', see Figure 5).

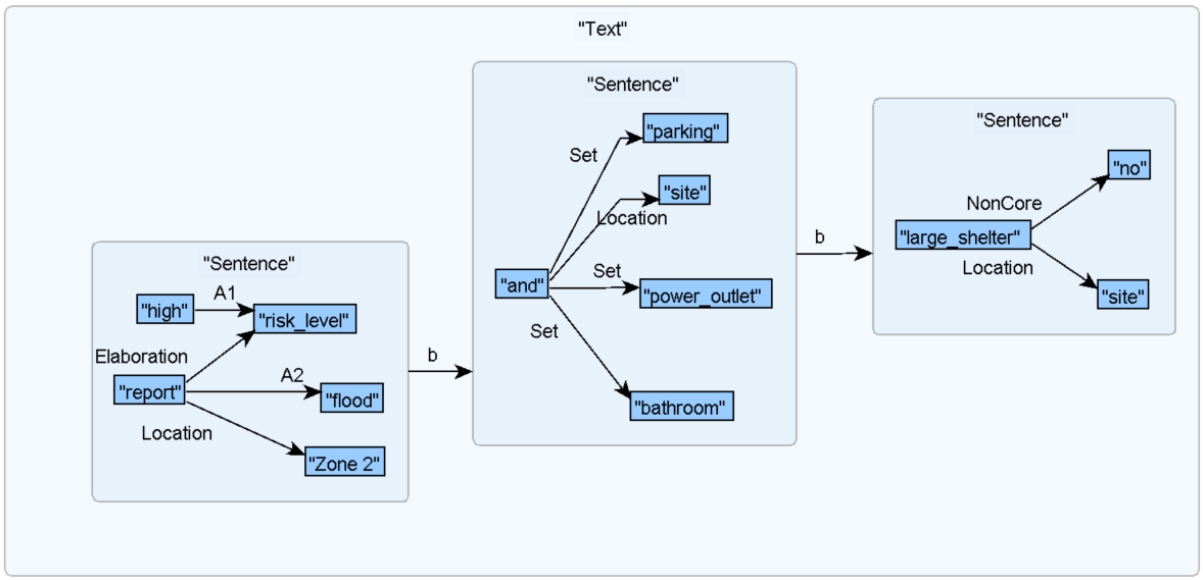

Figure 5: Sentence packaging through two-step aggregation of triples (step 1)

Second, we check if an argument of a predicate appears further down in the ordered list of discursive units. If so, the units are merged by fusing the common argument; during linguistic generation, this results in the introduction of postnominal modifiers such as relative and participial clauses or appositions (e.g., *A structure, which is a bridge, has been detected*). In order to avoid the formation of heavy nominal groups, we allow at most one aggregation by argument. Referring expressions are introduced during the next steps. Step 2 does not make any change to our running example.

One more action is performed during text planning, which is the determination of the communicative structure. In order to realise a sentence, it is necessary to give it a communicative orientation: What are we talking about? What do we say about it? The former is marked as theme of the sentence, the latter as rheme, based on the semantic relations and the nodes in each connected graph. Anything that does not belong to any of these two spans is by default a specifier. Each lexical unit is included in a communicative span (theme, rheme, specifier), which can contain any number of lexical units. For the three structures in Figure 5, the nodes 'report' and 'site' (twice) have been identified as being the main nodes of their respective sentences.

All the nodes have been assigned a part of speech and linked to an entry in a lexical resource: for instance, "report" is linked to the entry 'report_VB_01' according to the PropBank nomenclature (Kingsbury and Palmer 2002).

From Semantic Structure to Deep-Syntactic Structure (DSyntS): lexicalising and defining the sentence structure

During the transition from Semantic Structure to Deep-Syntactic Structure, the semantic graph with the communicative structure is mapped onto a tree: the main node of the rheme will be the head/root of the sentence, that is, the main verb, while the rest of the rheme generally corresponds to the objects and adverbs, and the theme to the syntactic subject. From the root, the whole tree is built node by node. The nodes are assigned a lexical label, in many cases one that is very close to the semantic label, but sometimes meanings map to more idiosyncratic words.

The aforementioned lexical resource indicates what a syntactic predicate requires in order to form a correct sentence in a language (syntactic combinatorial). For instance, the verb 'report', as most verbs, requires a noun or a non-finite verb as its subject. The subject may also have arguments, also restricted by the syntactic combinatorial.

Only meaningful units (lexical units) are part of the DSyntS; in other words, there are no grammatical units that lack semantic content at this point (bound prepositions, auxiliaries, etc.). The DSyntS can also contain abstract lexemes (collocates), formalised as Lexical Functions. Those Lexical Functions are given a value (a concrete label) during the DSyntS-SSyntS mapping (see next subsection) based on the combination with other words. For instance, the abstract lexeme Magn, which means 'a high degree of', would be realised as 'heavy' in combination with 'rain', but as 'deep' in combination with 'sleep'. In our running example, 'be' is introduced as a support verb for 'on site' to be realised as the main element in the sentence (prepositions generally cannot be the main element in a sentence).

Figure 7 shows that the main syntactic nodes of the sentences are the roots of the trees, and that all other elements are organised around each main node. Instead of a pure predicate-argument structure, the edge label reflects the syntactic structure of the sentence, in particular the opposition between arguments (I, II, IV) and modifiers (ATTR). Co-referring nodes are linked together with a blue-dotted line; these coreference links are used to introduce referring expressions (e.g., pronouns) in the next steps.

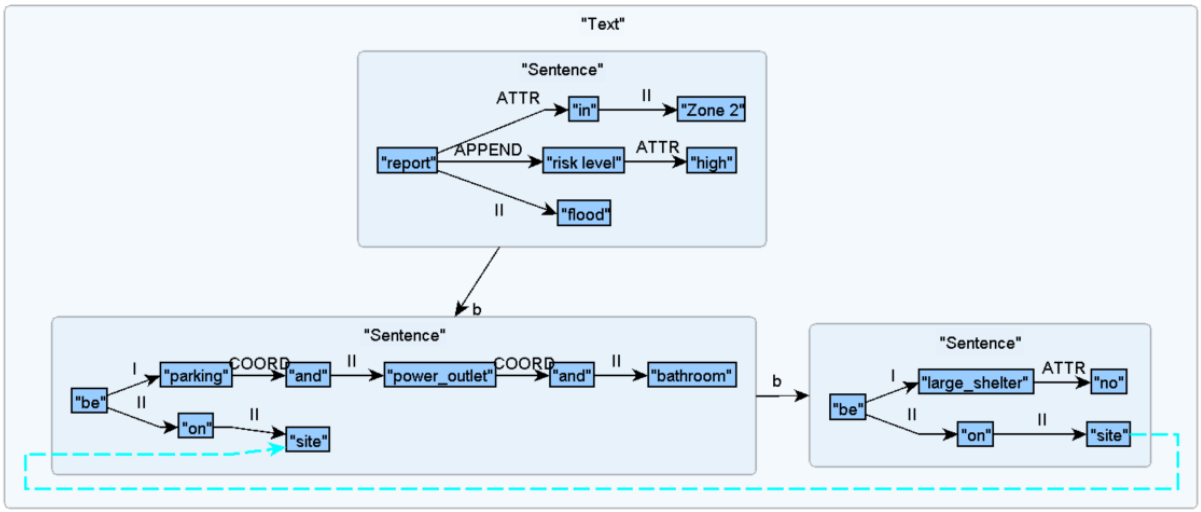

Figure 6: Deep-syntactic structures that correspond to the semantic structures in Figure 5

From Deep-Syntactic Structure to Surface-Syntactic Structure (SSyntS): introducing all idiosyncratic information

Once the structure of the sentence has been defined and all the meaningful words have been chosen, non-meaningful units need to be introduced. In the lexicon, an entry of a word indicates which preposition, case, finiteness, number, etc. has to be inserted on its dependent. For instance, the verb 'report' bears features that indicate the presence of the 'perfect' aspect (have + past participle), so the auxiliary 'have' is introduced in the SSyntS. Expletive subjects ('there') are also introduced when needed. Then other non-lexical nodes such as governed prepositions (e.g. 'of ' in 'source of light') and conjunctions, determiners, passive auxiliaries etc. are introduced. For instance, in Figure 7 the passive auxiliary 'be' is also introduced on 'report' in the first sentence in order to realise a deep-syntactic configuration that does not allow for generating in active voice (there is no first argument on the verb).

Furthermore, the generic syntactic relations found in DSyntS are refined into more idiosyncratic relations that convey very accurate syntactic information, instead of semantic (i.e., argument numbers). For instance, the DSyntS relation 'I' can be mapped to SBJ (subject) if the verb is active, OBJ (object) if the verb is passive, NMOD if the head is a noun, etc. A SBJ has the syntactic property to trigger an agreement on the verb, to undergo demotion in some conditions, and to be realised before the verb in a neutral sentence. An OBJ, on the contrary, appears by default after the verb, can undergo promotion, and is cliticisable with an accusative pronoun. An NMOD cannot be promoted or demoted, does not trigger any agreement, and always has to be realised to the right of its governor. Figure 7 shows the surface-syntactic structure with functional words and language-specific syntactic relations. Another separate

module takes care of possible elisions and pronominalisations, which avoid the generation of repetitive contents. For instance, the second occurrence of ‘on the site’ is more naturally rendered with an adverb ‘there’ in Figure 7.

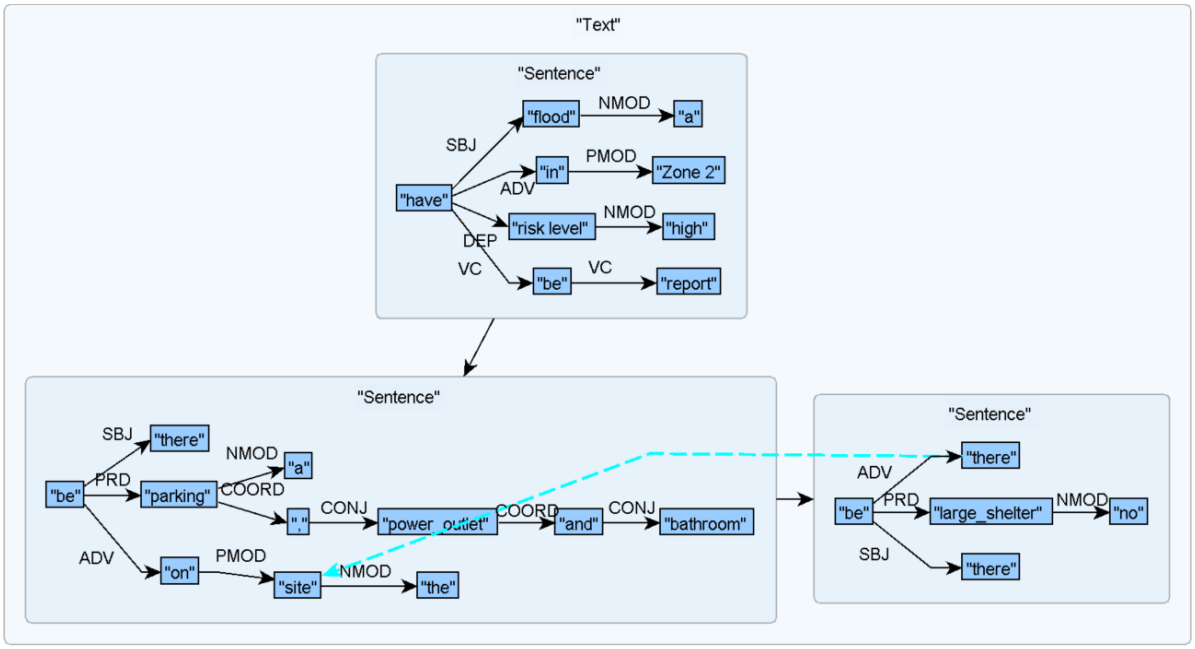

Figure 7: Surface-syntactic structures that correspond to the deep-syntactic structures in Figure 6

Finally, a submodule also performs syntactic aggregations, in order to combine together sentences that have elements in common that have not been aggregated in the deeper levels. In our example, the second and third sentences both have an expletive ‘there is’ construction, with a co-referring location. Some rules aggregate the two sentences together, with an ‘and’ or with a ‘but’ according to the properties of the PRD dependent. In this case, the first sentence has negation on the PRD, while the second one does, so in Figure 8, the last two sentences are coordinated with ‘but’.

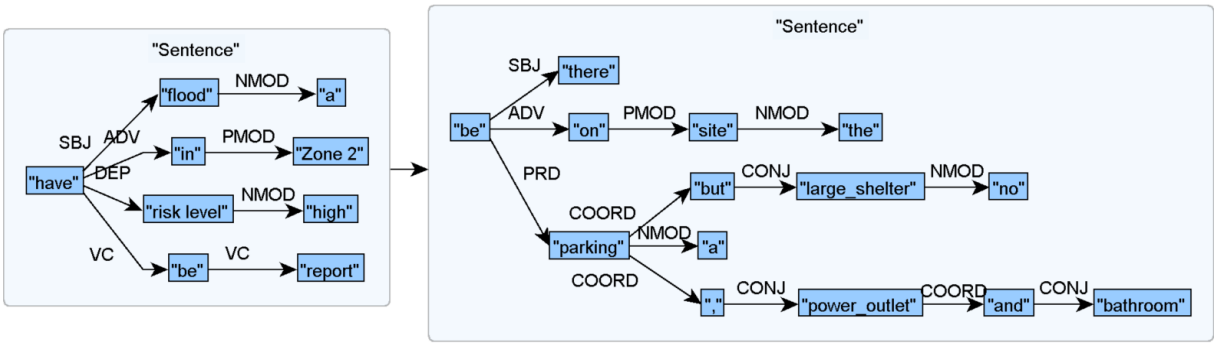

Figure 8: Surface-syntactic aggregation

From Surface-Syntactic Structure to Morphologic Structure (MorphS): resolving word agreements and word ordering

Thanks to the idiosyncratic set of surface-syntactic relations, all agreements between the components of the sentence can be resolved. Every word of the sentence contains all the indications to make the production of the final form possible. This can be done either by creating a full-fledged dictionary containing entries under the form, e.g., 'have<VB><IND><PRES><3><SG> = have, or by using some automata based on inflection schemas to automatically inflect forms. Capitalisation can be introduced when necessary.

Another advantage of using the idiosyncratic set of surface-syntactic relations is that the issue of order between the components of the sentence can be resolved effectively; for instance, in a given language, subject goes before its governing verb, a determiner before its governing noun, etc.

Figure 9 shows that, at this level, the words carry all the necessary information for inflection, i.e., part-of-speech, mood, tense, person, and number (the small window on the top left of the figure, which shows information related to 'have'). The precedence relations are in red.

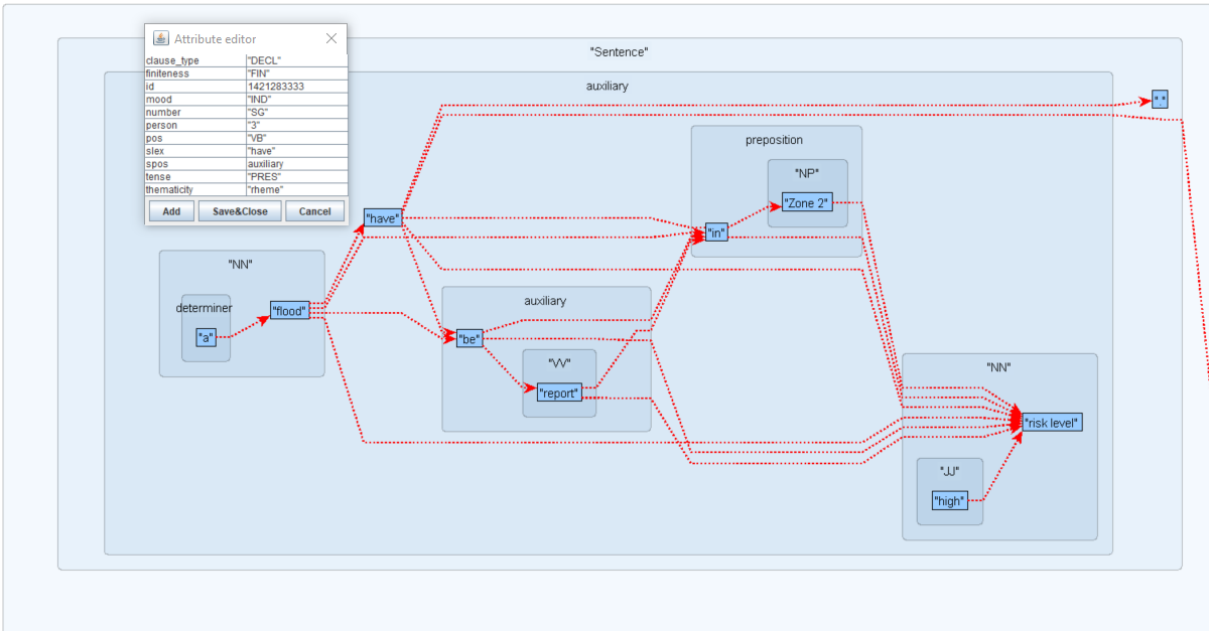

Figure 9: A (linearised) morphological structure that corresponds to the first surface-syntactic structure in Figure 8

From Morphologic Structure to Sentence: finalising the sentence

Once all the words are ordered, punctuation marks are introduced (periods and commas around descriptive modifiers), the final form of the words is retrieved, and the sentence is ready to be delivered to the next module. In the case of the running example shown throughout this section, the output would be the following:

A flood has been reported in Zone 2 (high risk level). There is a parking, power outlets and bathrooms but no large shelters on the site.

4.2.2 Implementation with the UPF-FORGe generator

The basic text generation implementation consists of manually crafted graph-transduction grammars for each transition between two consecutive layers. In combination with the rules, dictionaries of two different types are required: one that describes the syntactic properties of these words (lexical dictionary), and one that contains the inflection patterns of each word (morphological dictionary). The sample entry in Figure 10 shows the syntactic properties of the verb 'report', which has three nominal arguments in government patterns with the third one being introduced by the preposition 'to' (to report something to someone).

```
"report_VB_01":_verbExtArg_{
 lemma = "report"
 gp = {
        I = {
                pos = NN
                 rel = SBJ
         }
        II = {
                pos = NN
                rel = OBJ
         }
         III = {
                pos = NN
                rel = IOBJ
                prep = "to"
         }
    }
}
```

Figure 10: Syntactic properties of the verb 'report'

The generation module will cover all languages involved in interactions with the Knowledge Base in the project with different coverage, directly related with the size of the respective lexicons.

For the implementation of the graph-transduction rules that map the conceptual structures onto text, we have been re-implementing the graph transducer MATE (Bohnet and Wanner, 2010) in order for the transduction rules to be more expressive and compact, as well as for the tool to perform the transductions faster. We will refer to this new tool as "MATE-2" in this section.

MATE is a graph transducer programmed in Java. It contains different editors for graph construction, rule and lexical resource writing, a debugger, as well as a tool for regression tests. The rules (and their corresponding conditions) match a part of an input graph, and create a part of the output graph. The main problem with MATE is its speed, which was not adequate for a real-time system as needed in xR4DRAMA. MATE-2 is currently in an advanced state but no publication describes it yet. Figure 11 shows a project open in MATE-2.

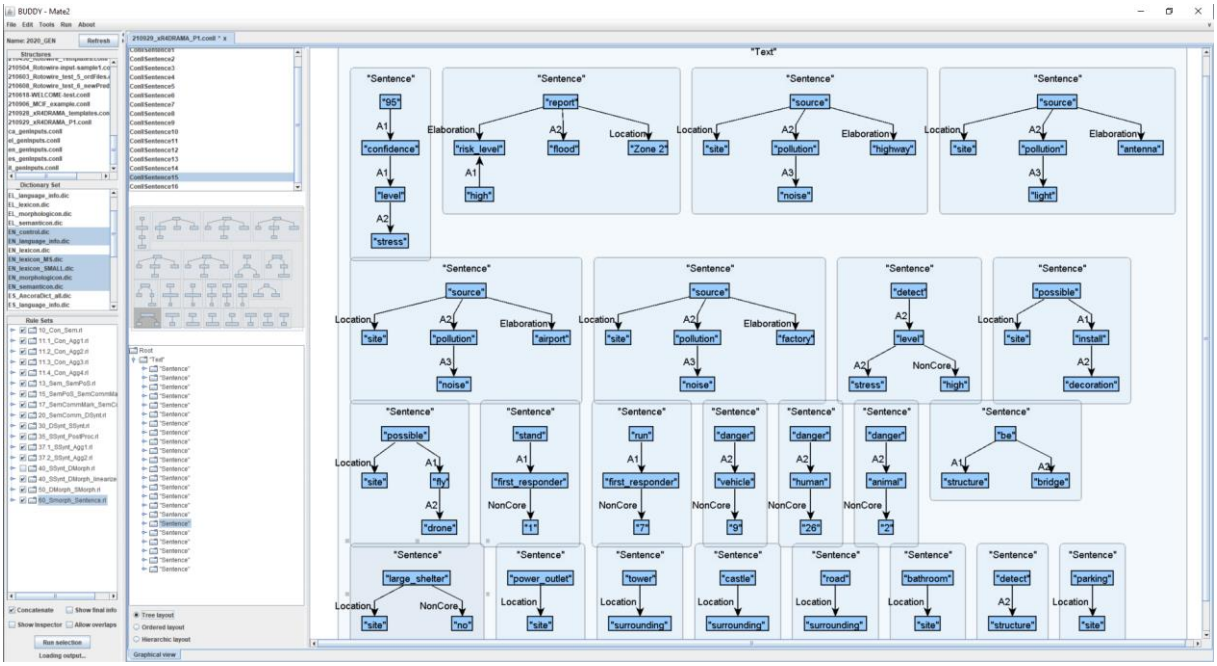

Figure 11: A screenshot of MATE-2 (Graph Editor view)

MATE-2 contains:

- A **Project Browser**, used to open and navigate in the different resources of a project. It is the column on the left end of the screenshot, which contains 3 list fields, some options and the Run Selection button. Each list field corresponds to a resource used for the transduction:
  - Structures: a list of input structures on which the rules can be applied in order to create new structures;
  - Dictionary Set: a list of lexical resources used by the rules;
  - Rule Sets: a list of rule sets (= grammars); each rule set performs one transduction.
- A **Resource Editor**, for opening each of the three resources in an editor tab, by double clicking on it. It occupies the main part of the screen, on the right of the Project Browser.
  - The **Graph editor** contains 5 fields (see Figure 11):
    - Graph List: the list of graphs contained in one file (top left);
    - Graph Global View: the global view of the selected graph (middle top left);
    - Graph Node List: the list of nodes of the selected graph (middle bottom left);
    - Graph layout options (bottom left);
    - Graph View: the complete graph view of the selected graph (right).
  - The **Rule editor** contains 2 main fields and some parameters (see Figure 12):
    - Rule List: the rule tree (left);
    - Rule View: the complete rule view of the rule selected in the rule tree (right).

    The Rule editor is user-friendly with different styles and colours showing the different component of the input (Left Side) and output (Right Side) graphs: logical operators are shown in **orange**, parentheses, round brackets (which delimitate nodes) and relation markers in **red**, quoted strings in **light blue** or

normal strings in black, variables in **dark blue**, dictionary references in **bold black**, and comments in ***italics green***. Matching parentheses and backets can be highlighted.

- The **Lexicon editor**'s advanced view is still quite simple, but shows clearly the information stored for each lexical unit (see the entry for 'report' in Figure 13):
  - Entry list and hierarchy (left);
  - Summary of classes an entry inherits features from (top right);
  - Details of the entry, in particular collocation and subcategorisation information (bottom right).

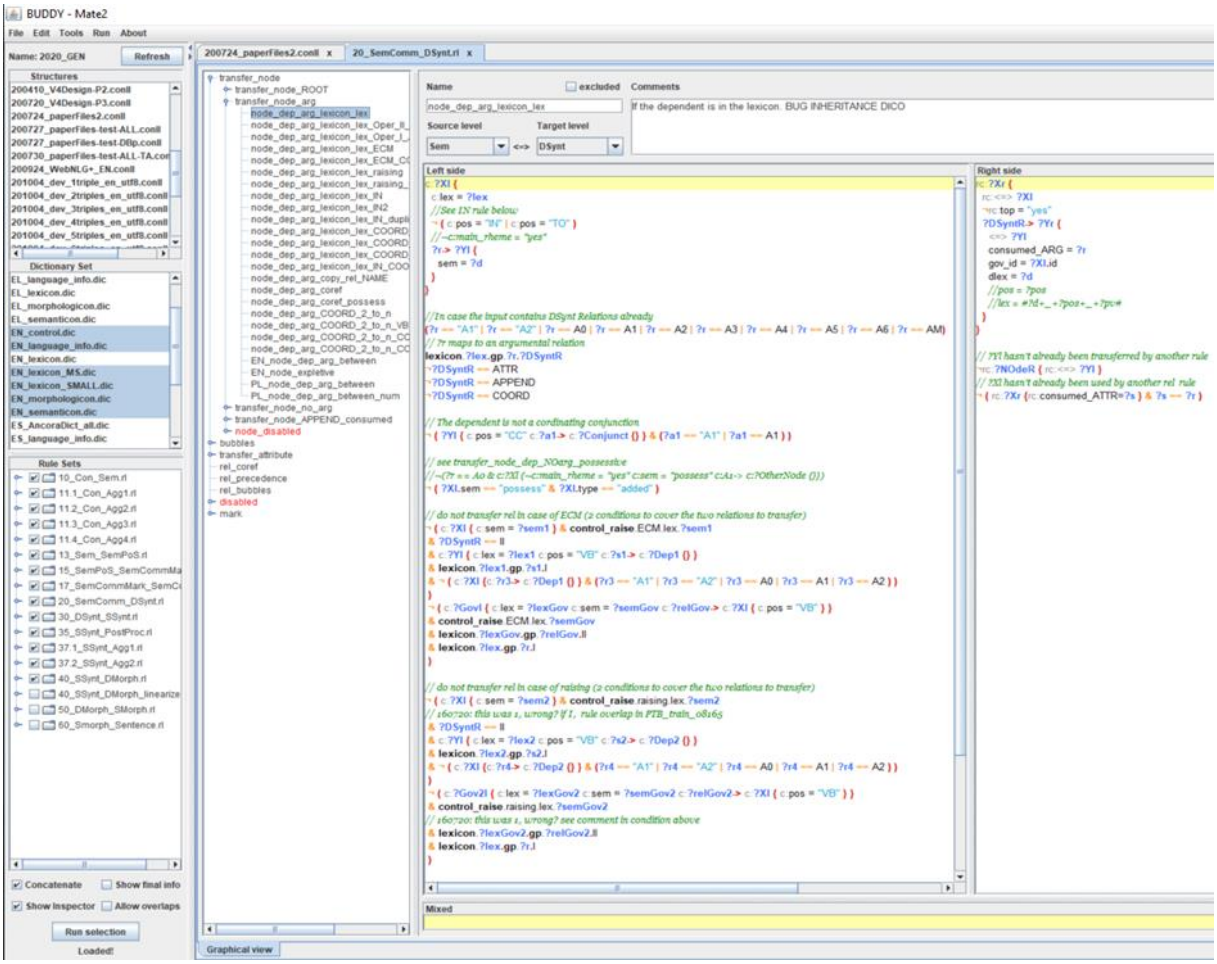

Figure 12: A screenshot of MATE-2 (Rule Editor view)

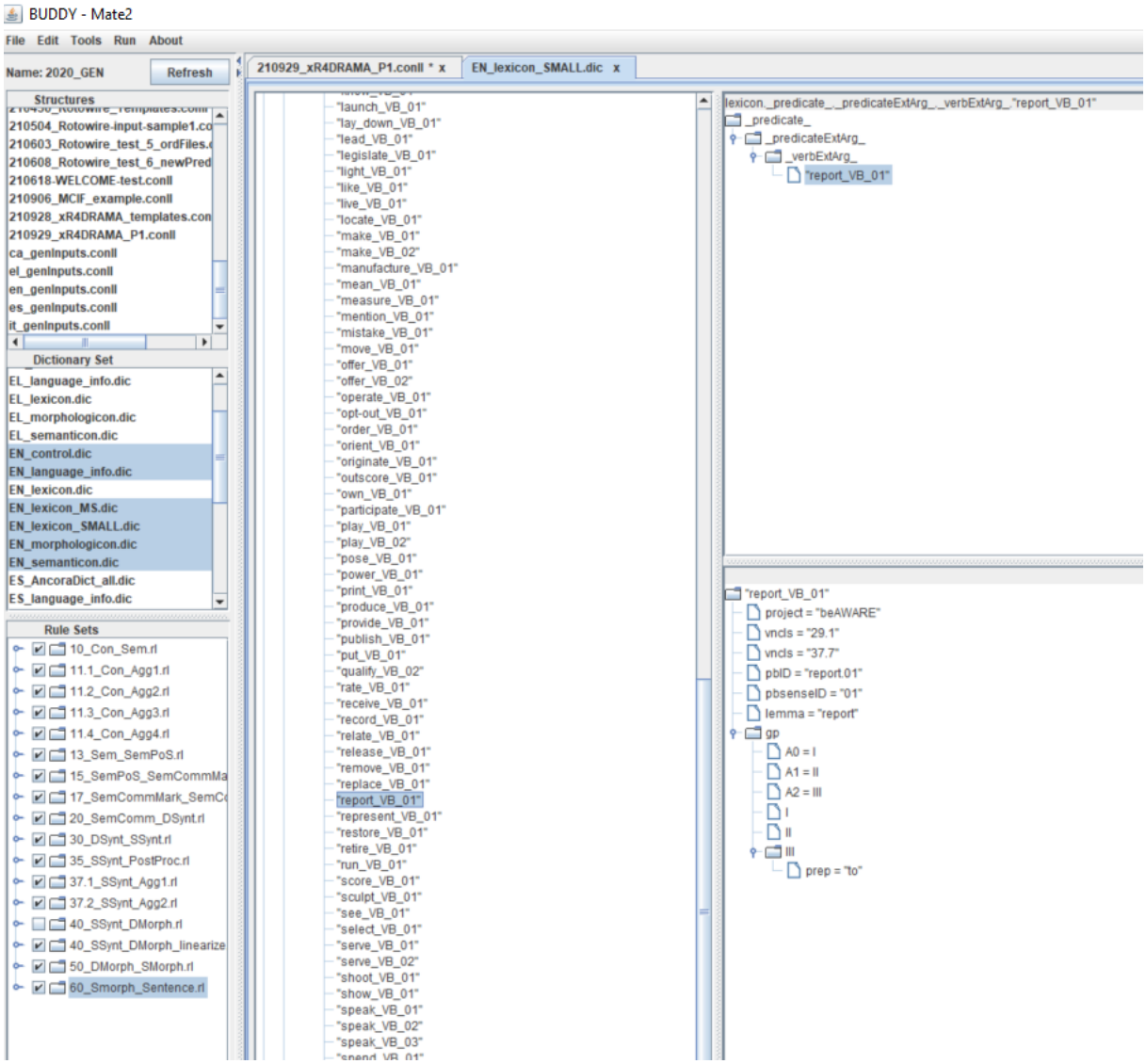

Figure 13: A screenshot of MATE-2 (Lexicon Editor view)

Finally, both for debugging and demonstration purposes, MATE-2 also has an execution view, called *Inspector*, in which we can trace how each rule applied to the input graph, and the output graph is incrementally built. In Figure 14 below, the project browser is on the left-hand side, and the rest of the screen shows the inspector: a column containing (from top to bottom) input graph name, executed grammars, executed rule clusters within the grammar, which rules were applied in one cluster, the instances of these rules, and how the variables from the rule on the bottom right of the figure were instantiated on the input graph shown at the top. The output graph is also shown at the top, with green arrows linking each node of the output to its origin in the input.

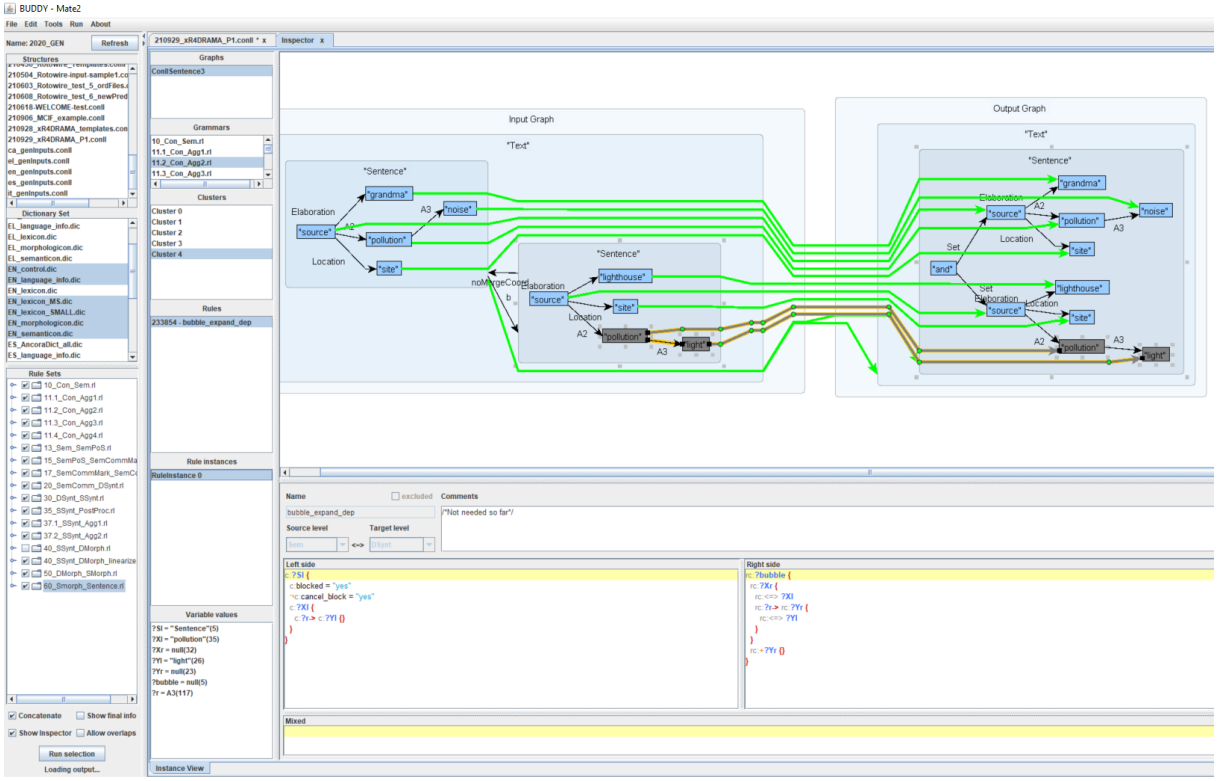

Figure 14: Inspector view of MATE-2

5 IMPROVEMENT TO TOOLS, DATASETS AND METHODS

In this section, we describe both the project-specific improvements and the improvements to tools, datasets and evaluation methods that will have an impact and use beyond xR4DRAMA.

5.1 Grammar-based generation

During the first year of the project, we extended the coverage of UPF's grammar-based generator FORGe (Mille et al. 2019) so as to (i) address the requirements of xR4DRAMA, and (ii) improve UPF's generator's quality and portability in general.

For (i), we built a series of simulated inputs for the xR4DRAMA Use Cases and ensured the robustness of the generator across the different input configurations. In particular, we (i.i) crafted the predicate/argument templates needed to cover the data points required by the Users, (i.ii) extended our lexical resources to describe the lexical units used in the PredArg templates, (i.iii) developed some project-specific aggregation rules (semantic and syntactic aggregation, see Section 4.2) to verbalise more naturally multiple sentences with the same location, and (i.iv) developed some generic rules for handling specific types of elisions and pronominalisations to further avoid repetitions in the output text.

For (ii), we kept improving our generator on the most challenging benchmark dataset for structured data-to-text natural language generation, WebNLG (Castro Ferreira et al., 2020). FORGe already got excellent results on the WebNLG dataset, but human evaluations showed some limitations of FORGe with respect to the fluency of the generated language. We implemented new generic aggregation rules (semantic and syntactic levels) to produce more diverse and natural aggregations, and implemented a new morphology generation module to increase the robustness of the generator to new inputs. Some of the suboptimal outputs were improved (see examples in Table 9) but we did not carry out a global qualitative evaluation of the generator during the first year of xR4DRAMA. Note that all improvements made on this benchmark data also benefit the xR4DRAMA reports; working on WebNLG allows us to ensure the quality of the grammars on a wide variety of inputs.

xR4DRAMA M0	**xR4DRAMA M12 (improvements in blue)**
Atatürk_Monument_(İzmir) , inaugurated on July_27_(1932) , is made of Bronze . it is in Turkey , the leader of which is Ahmet_Davutoğlu . the capital of Turkey is Ankara . the largest city in Turkey is Istanbul . the currency of Turkey is the Turkish_lira .	Atatürk_Monument_(İzmir) , inaugurated on July_27_(1932) , is made of Bronze . it is in Turkey , the leader of which is Ahmet_Davutoğlu . the capital of Turkey is Ankara , the largest city in Turkey Istanbul and the currency of Turkey the Turkish_lira .
William_Anders , who was selected by NASA in 1963 , retired on September_01,_1969 . he spent 8820_minutes in space and is a fighter_pilot . he was born in British_Hong_Kong on October_17,_1933 . he was a crew member of Apollo_8 .	William_Anders , selected by NASA in 1963 , retired on September_01,_1969 . he spent 8820_minutes in space , was born in British_Hong_Kong on October_17,_1933 and is a fighter_pilot . he was a crew member of Apollo_8 .
the_character_Bolt , which Gary_Cohn and Dan_Mishkin created , is also known as Larry_Bolatinsky .	the_character_Bolt , created by Gary_Cohn and Dan_Mishkin , is also known as Larry_Bolatinsky .

Table 9: Improvements on sample raw outputs of the FORGe generator (WebNLG dataset)

Table 10 summarises the development of the grammars during the course of the first year of xR4DRAMA, taking as starting point the generator as reported in November 2020 in the final deliverable of the V4Design project (D5.5, H2020-779962), i.e., just before the start of xR4DRAMA.

	xR4DRAMA M0	**xR4DRAMA M12**
Languages supported	-	EN
Number of rules	1,995	2,147
% of language - independent rules	Con-SMorph (1,995) : 74%	Con-SMorph (2,147) : 75%
	1 - Con-Sem (468) : 97% 2 - Aggregation (309) : 100% 3 - Sem-DSynt (232) : 78% 4 - DSynt-SSynt (645) : 56% 5 - SSynt-DMorph (223) : 53% 6 - DMorph-SMorph (117) : 50%	1 - Con-Sem (505) : 97% 2 - Aggregation (357) : 100% 3 - Sem-DSynt (243) : 78% 4 - DSynt-SSynt (670) : 56% 5 - SSynt-DMorph (241) : 54% 6 - DMorph-SMorph (131) : 54%

Table 10: Overview of the development of the generation grammars

5.2 Neural generation

We have been experimenting with the WebNLG reference dataset and trying to answer the following question: is it possible to improve FORGe's fluency without sacrificing its accuracy in terms of contents, and get texts of better quality than those produced by the best neural generators?

To explore this, we are (i) applying a neural text paraphrasing tool to the outputs of FORGe and (ii) developing methods to choose when to apply the paraphrasing and when not to. The idea is to obtain paraphrased (usually more fluent) outputs whenever possible (i.e., whenever we are sure it does not harm the accuracy). In the case that paraphrasing is not an option, because some omissions or hallucinations are detected, we fall back to the FORGe outputs. This way, we aim at further improving the results of the human evaluation of FORGe' outputs, including in the context of xR4DRAMA. If the experiments are successful, we will report on them in the final deliverable of xR4DRAMA.

5.3 Datasets

Data augmentation is an important component in the robustness evaluation of models in natural language processing (NLP) and in enhancing the diversity of the data they are trained on. We actively participated in the design, creation and release of NL-Augmenter, a new participatory Python-based natural language augmentation framework which supports the creation of both transformations (modifications to the data) and filters (data splits according to specific features). We released an initial dataset of 117 transformations and 23 filters for a variety of natural language tasks, and we demonstrated the efficacy of NL-Augmenter by

measuring the capability of some of the transformations for analyzing the robustness of multiple popular natural language models, mostly in the Natural Language Generation domain. The infrastructure, datacards and robustness analysis results are available publicly on the NL-Augmenter repository: https://github.com/GEM-benchmark/NL-Augmenter.

5.4 Evaluation

In addition to creating new datasets for the evaluation of (among others) NLG systems, we carried out experiments about their evaluation, first by developing a new state-of-the-art accuracy error detection tool, and second by studying the degree of reproducibility of human evaluations of NLG systems.

Automatic error detection in NLG outputs

We developed an automatic metric for token-level error annotation which combines a rule-based generation system (xR4DRAMA's FORGe system) with a neural retrieval model and a pretrained neural LM used for error tagging. Figure 15 shows the types of text generated by FORGe (right) and a simple Python-based template generator (basketball dataset), while Figure 16 shows an overview of the whole system.

Figure 15: Rule-based NLG which we use to generate facts (data points) from the input data. The facts are used as an input to the error checking model. We experimented with (a) simple hand-crafted templates and (b) compact sentences generated by the FORGe system.

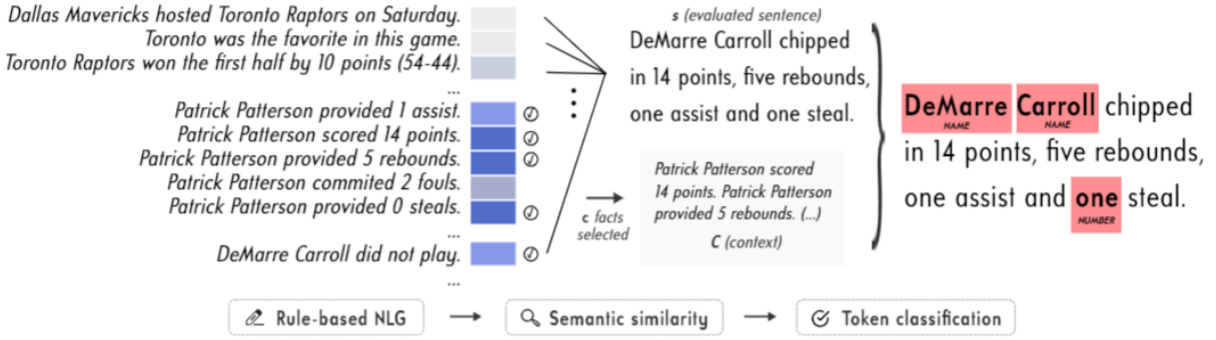

Figure 16: An overview of our system. First, we generate the facts from the input table with a rule-based NLG system (see Figure 15). For each evaluated sentence s, we select c facts with the highest semantic similarity, getting a context C. The pair (C, s) is given as an input to a pre-trained LM for token-level error classification.

We evaluated our approach in a cross-validation scenario to select the best configuration for the shared task. Overall, our system is able to reach a 65% error detection F1 score and ranked

first out of four automatic submissions in the Shared Task on Evaluating Accuracy in Generated Texts at INLG 2021 (Reiter and Thompson, 2021). The code for our experiments is freely available on Github: https://github.com/kasnerz/accuracySharedTask_CUNI-UPF.

Reproducibility

Recent years have seen growing interest in, and concern about, reproducibility across the Natural Language Processing (NLP) field. The evaluation of NLG systems as will be performed in xR4DRAMA is very challenging, and controlling the level of reproducibility of an evaluation may allow for obtaining more solid results. The ReproGen Shared Task on Reproducibility of Human Evaluations in Natural Language Generation (Belz et al., 2021) was the first shared task to focus on reproducibility of human evaluations (rather than metrics). We participated in ReproGen, where our contribution was in Track A, the Main Reproducibility Track. More specifically, we repeated the human evaluation study reported by (van der Lee et al., 2017).

The human evaluation studied here is about as simple as such evaluations get: just one system was evaluated, on three quality criteria (Fluency, Clarity, stance identification) and 10 output pairs, each evaluated by the same 20 raters. The coefficient of variation we used gives a measure of degree of reproducibility that is comparable across measures and across studies, so we can e.g., make the (relative) assessment that Clarity was found to have a higher degree of reproducibility than Fluency. However, the measure does not enable us to make an (absolute) assessment whether either one of them had *good* reproducibility. In order to do this, we would have to know what normally counts as good reproducibility in similar circumstances in NLP. Since NLP currently has very few reproduction studies, and none that report coefficients of variation for human evaluations, such assessments are not possible at this point in time. They will become possible over time if more studies start to report CV (or other measures of precision) for reproduction studies, which is why we believe that our effort is crucial for the future of human evaluation of generation systems.

6 CONCLUSIONS & WORK FOR SECOND YEAR

This document presents the progress attained in the first half of the project with respect to the task of multilingual generation in WP3. The state of the art at the beginning of the project is compared to the recent achievements that show improvements in the central subtasks which were done in accordance with the project timeline. Target texts to be generated have been defined with the user partners, and a first interface with the Knowledge Base has been defined.

Five main improvements were made to the UPF FORGe multilingual discourse generator:

- initial predicate/argument templates and lexical resources were crafted to cover the KB data points to be verbalised in xR4DRAMA;
- rules were extended to cover project-specific types of sentence packaging so as to make the generated texts more fluent; the general coverage of the rules was improved: all grammars are now more complete, with 2,147 rules in total, as opposed to 1,995 at the beginning of the project;
- a new morphology generation module has been implemented;
- some rules were generalised, and made more language independent: now, 75% of the rules are language-independent, as opposed to 74% at the beginning of the project;
- updated rules caused the significant improvement of the quality of the English generator on a challenging benchmark dataset (WebNLG);

In addition, new datasets for training and testing statistical generation tools have been developed and released publicly, and some experiments combining neural and grammar-based methods have been undertaken, and methodologies for evaluating the outputs of generation systems have been proposed and validated in the context of a couple of international shared tasks.

The xR4DRAMA use cases are currently covered in accordance with the progress in the definition of the two use cases.

In the scope of future work, we will aim to improve the coverage for Italian and German, and improve the quality of English. We will particularly work closely with the KB to ensure a smooth communication between the two modules and cover all the contents needed by the project. Further experiments with deep learning techniques will be carried out.

Four publications in the context of xR4DRAMA were produced:

Kasner, Z., S. Mille and O. Dušek (2021). Text-in-Context: Token-Level Error Detection for Table-to-TextGeneration. In *Proceedings of the 14th International Conference on Natural Language Generation*, pp. 259-265, Aberdeen, UK (Online). **pdf**

Mille, S., T. Castro-Ferreira, A. Belz and B. Davis (2021). Another PASS: A Reproduction Study of the Human Evaluation of a Football Report Generation System. In *Proceedings of the 14th International Conference on Natural Language Generation*, pp. 286-292, Aberdeen, UK (Online). **pdf**

Mille, S., Dhole, K.D., Mahamood, S., Perez-Beltrachini, L., Gangal, V., Kale, M., van Miltenburg, E. and Gehrmann, S. (2021). Automatic Construction of Evaluation Suites for Natural Language

Generation Datasets. In *Proceedings of the Thirty-Fifth Annual Conference on Neural Information Processing Systems, Datasets and Benchmarks Track (Round 1)*. In Press.

Pérez-Mayos, L., A. Táboas García, S. Mille and L. Wanner (2021). Assessing the Syntactic Capabilities of Transformer-based Multilingual Language Models. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3799-3812. Association for Computational Linguistics. **pdf**

7 REFERENCES

Androutsopoulos, I., Lampouras, G. and D. Galanis. 2013 "Generating natural language descriptions from OWL ontologies: the NaturalOWL system." In *Journal of Artificial Intelligence Research* 48, pp. 671-715.

Ballesteros, M. et al., 2015. "Data-driven sentence generation with non-isomorphic trees". In Proceedings of NAACL-HLT, pp. 387-397.

Belz, A., Shimorina, A., Agarwal, S. and Reiter, E., 2021, August. The ReproGen Shared Task on Reproducibility of Human Evaluations in NLG: Overview and Results. In *The 14th International Conference on Natural Language Generation*.

Bohnet, B. et Wanner, 2010. "Open-Source Graph Transducer Interpreter and Grammar Development Environment". In *Proceedings of LREC*, pp.211-218.

Castro Ferreira, T., Gardent, C., Ilinykh, N., van der Lee, C., Mille, S., Moussallem, D. and Shimorina, A. 2020. "The 2020 bilingual, bi-directional webnlg+ shared task overview and evaluation results (WebNLG + 2020)". In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*.

Chen, Z. et al., 2020. "Few-Shot NLG with Pre-Trained Language Model". In *Proceedings of ACL*, pp.183-190.

Gardent, C., et al., 2017. "Creating training corpora for nlg micro-planning". In *Proceedings of ACL*, pp.179-188.

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

Gatt, A. and Reiter, E., 2009. "SimpleNLG: A realisation engine for practical applications". In *Proceedings of ENLG*, pp. 90-93.

Kingsbury, P. and Palmer, M., 2002. "From TreeBank to PropBank". In *Proceedings of LREC*, pp. 1989-1993.

Mel'čuk, I.A., 1988. *Dependency syntax: theory and practice*. SUNY press.

Meyers, A., et al, 2004. "The NomBank project: An interim report". In *Proceedings of the Workshop Frontiers in Corpus Annotation.*

Mille, S., Dasiopoulou, S. and Wanner, L., 2019. "A portable grammar-based NLG system for verbalization of structured data". In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing,* pp. 1054-1056.

Moussallem, D., et al. 2018. "RDF2PT: Generating Brazilian Portuguese Texts from RDF Data". arXiv preprint arXiv:1802.08150.

Nayak, et al. 2017. "To plan or not to plan? discourse planning in slot-value informed seq2seq models for LG". In *Proceedings of Interspeech*.

O'Donnell, Mick, Chris Mellish, Jon Oberlander, and Alistair Knott. 2001. "ILEX: an architecture for a dynamic hypertext generation system". In *Natural Language Engineering* 7, no. 3: 225.

Perez-Beltrachini, Laura, Rania Sayed, and Claire Gardent. 2016. “Building rdf content for data-to-text generation”, In *Proceedings of The 26th International Conference on Computational Linguistics (COLING 2016)*, Osaka, Japan.

Rambow, O. and Korelsky, T. 1992. "Applied text generation". In *Proceedings of ANLP*, ACL, pp.40-47.

Reiter, E. and Thomson, C., 2020, December. Shared Task on Evaluating Accuracy. In *Proceedings of the 13th International Conference on Natural Language Generation* (pp. 227-231).

Schuler, K.K., 2005. "VerbNet: A broad-coverage, comprehensive verb lexicon".

van der Lee, C., Krahmer, E. and Wubben, S., 2017, September. PASS: A Dutch data-to-text system for soccer, targeted towards specific audiences. In *Proceedings of the 10th International Conference on Natural Language Generation* (pp. 95-104).

Walter, S., C. Unger, and P. Cimiano. 2014. “M-ATOLL: A Framework for the Lexicalization of Ontologies in Multiple Languages”, In *Proceedings of Semantic Web Conference*. Riva.

Wanner, L., Bohnet, B., Bouayad-Agha, N., Lareau, F. and D. Nicklaß. 2010. "MARQUIS: Generation of user-tailored multilingual air quality bulletins." In *Applied Artificial Intelligence* 24, no. 10, pp. 914-952.

Zhao, C. et al., 2020. "Bridging the structural gap between encoding and decoding for data-to-text generation". In *Proceedings of ACL*, pp. 2581-2491.