V4Design

Visual and textual content re-purposing FOR(4) architecture, Design and virtual reality games

H2020-779962

D5.2

Basic Summarisation Techniques

Dissemination level:	Public
Contractual date of delivery:	Month 16, 30/04/2019
Actual date of delivery:	Month 17, 02/05/2019
Work Package:	WP5 Content integration, retrieval and presentation
Task:	T5.4 Ontology-based text planning T5.5 Multilingual explanatory text generation for the 3D objects
Type:	Report
Approval Status:	Final
Version:	1.1
Number of pages:	47
Filename:	D5.2_V4Design_Basicsummarizationtechniques_20190502_v1.1.docx

Abstract

This document describes the progress of tasks T5.4 (Ontology-based text planning) and T5.5 (Multilingual explanatory text generation for the 3D objects). This deliverable contains: (i) the state of the art in the considered fields of summarisation and multilingual text generation, (ii) a description of the basic approaches and respective implementations undertaken in V4Design for the two tasks, including, as part of text generation, an advanced module for sentence packaging, (iii) the results of evaluations of the carried out implementations, and (iv) the plans towards the realisation of the advanced techniques for summarisation and text generation.

given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

co-funded by the European Union

History

Version	Date	Reason	Revised by
0.1	26/02/2019	ToC	Gerard Casamayor
0.2	9/04/2019	Contribution to SoA and text planning sections.	Gerard Casamayor
0.3	15/04/2019	Contributions to text generation sections	Simon Mille
0.4	17/04/2019	Contributions to advanced sentence packaging section	Alexander Shvets
0.5	18/04/2019	Evaluation of text planning section	Gerard Casamayor
0.6	22/04/2019	Integration of different sections	Simon Mille
1.0	24/04/2019	Final version for internal review.	Simon Mille
1.1	01/05/2019	Version addressing internal review by CERTH	Simon Mille

Author list

Organisation	Name	Contact Information
UPF	Gerard Casamayor	gerard.casamayor@upf.edu
UPF	Simon Mille	simon.mille@upf.edu
UPF	Aleksander Shvets	alexander.shvets@upf.edu

Executive Summary

This document describes the progress of WP5 in the V4Design project. It focuses on tasks T5.4 (Ontology-based text planning) and T5.5 (Multilingual explanatory text generation for the 3D objects). These two tasks are responsible for verbalising, in the language of choice of the end user, the structured data (metadata of images and videos, aesthetic features of buildings and objects, etc.) that ends up in the Knowledge Base.

This deliverable contains: (i) the state of the art in the considered fields of summarisation and multilingual text generation, (ii) a description of the basic approaches and respective implementations undertaken in V4Design for the two tasks, including, as part of text generation, an advanced module for sentence packaging, (iii) the results of evaluations of the carried out implementations, and (iv) the plans towards the realisation of the advanced techniques for summarisation and text generation.

Both the summarisation and the text generation components are being developed as expected, and their respective evaluations show very promising results. Extractive summarisation algorithms and the basic text generation pipeline have been integrated successfully in the V4Design architecture.

Abbreviations and Acronyms

AMR	Abstract Meaning Representation
BFS	BabelNet First Sense
BoW	Bag-of-Words
ConS	Conceptual Structure
DnS	Description and Situation
DoA	Description of Action
DSyntS	Deep-Syntactic Structure
dul	DOLCE+DnS Ultralite
EL	Entity Linking
FORGe	Fabra Open-source Rule-based Generator
IE	Information Extraction
KB	Knowledge Base
LG	Language Generation
MAP	Mean Average Precision
MorphS	Morphological Structure
MTT	Meaning-Text Theory
NE	Named Entity
NER	Named Entity Recognition
NLG	Natural Language Generation
NLP	Natural Language Processing
OIE	Open Information Extraction
OOV	Out-of-Vocabulary
PCA	Principal Component Analysis
RDF	Resource Description Format
RST	Rhetorical Structure Theory
SemS	Semantic Structure
SRL	Semantic Role Labelling
SSyntS	Surface-Syntactic Structure
SW	Semantic Web
VR	Virtual Reality
WSD	Word Sense Disambiguation

Table of Contents

1 INTRODUCTION

In V4Design, the role of T5.4 (Ontology-based text planning) and T5.5 (Multilingual explanatory text generation for the 3D objects) is to communicate in a natural way non-human-friendly metadata to end users (Natural Language Generation) and allow them to have a quick grasp of large quantities of text (Summarisation). During the first half of the project, we followed a basic summarisation paradigm consisting of the identification of most relevant text chunks was followed (*extractive summarisation*), which will serve as a basis for the advanced implementation. For text generation, the consortium focused on the techniques for the automatic description of the 3D objects. More specifically, after consultation with the user partners, it has been decided that the descriptions should contain (at least) the following information:

- Metadata about the original material (videos or images): quality, size of file, frames per second, date of shooting, author of image/video, bit rate, etc.
- Features of the reconstructed 3D object: name, nickname, localisation (type of building/object), origin/part_of, date of construction, geolocation/location, architect/designer/creator, style.

The metadata of the original material comes with said material (images and videos), while the features of the buildings or objects will need to be extracted from images, videos, or texts. All the data will end up in the Knowledge Base (KB), and the relevant contents will be sent to the Text Generation pipeline for being verbalised in the language of choice of the end user.

In this deliverable, we present techniques for the verbalisation of the V4Design Knowledge Base, and for the summarisation of the contents to be put forward in a text, through extractive and abstractive techniques. The extractive summarisation techniques (realised as *text planning*) are combined with the text generation module and the text analysis module in order to make a so-called abstractive summarisation pipeline, which will aim at rewriting a certain amount of text in a more concise way (see Figure 1). Although Word Sense Disambiguation (WSD) and Entity Linking (EL) also belong to WP3 (Text Analysis), we address them jointly within the text planning tasks of WP5 and they are therefore also addressed in this deliverable.

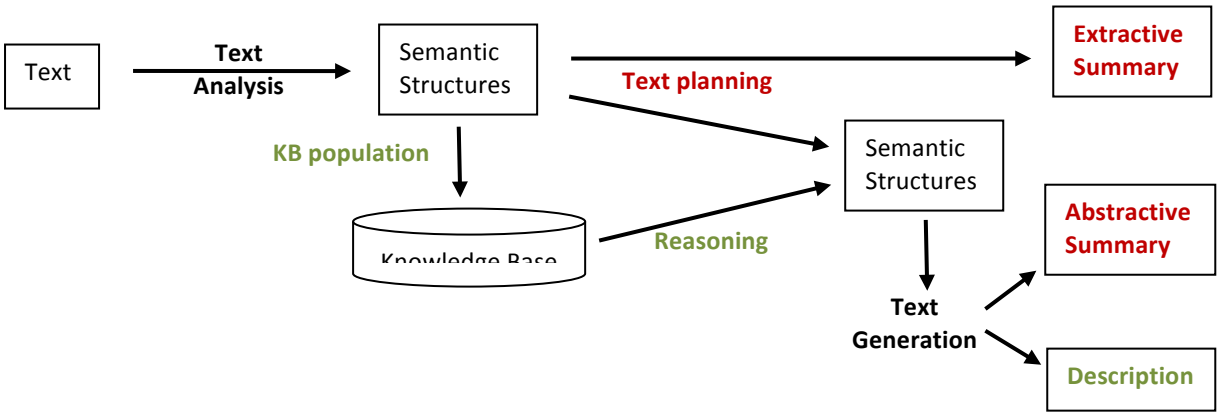


Figure 1: The summarisation components (text planning and text generation) in the context of V4Design

To cover the description of the above-mentioned tasks, the rest of the document is structured as follows. Section 2 contains the state of the art in summarisation and multilingual text generation. Sections 3 and 4 describe the basic techniques for summarisation and multilingual text generation, respectively. Section 5 reports on the plans and achievements with respect to the advanced techniques for both tasks. Finally, Section 6 summarises the document.

The whole work on summarisation (Section 3) and the advanced techniques (Section 5) has been done during the time frame of V4Design. However, for text generation, we built on work previously done in other EU projects. Section 4.10 clarifies what has been done during this project.

2 STATE OF THE ART

In this section, we first present the state of the art in abstractive summarisation (T5.4), and then in multilingual text generation (T5.5).

2.1 Abstractive summarisation

The traditional paradigm for abstractive summarisation includes building an intermediate representation through text analysis methods that serves as the basis for selecting the contents and rendering them as text using natural text generation methods. Works following this paradigm can be classified according to the depth of the analysis and the level of abstraction of the resulting representation. More recent approaches based on trained neural models address summarisation as a paraphrasing task where the sequence of words in the source text is mapped to the sequence of words that make up the summary. This new paradigm uses sequence to sequence methods to bypass the need for an intermediate representation.

2.2 Approaches based on deep analysis

By deep analysis we mean approaches that perform a linguistic analysis of the source document that goes beyond syntax, or alternatively apply information extraction methods to build a non-linguistic representation. Summarisation based on deep linguistic analysis typically relies on deep syntactic, Semantic Role Labelling (SRL) and semantic parsing tools to gain an understanding of the material to be summarised.

An example of this approach is (Khan, et al. 2015), where predicate-argument tuples are extracted from documents using an SRL tool. The tuples are clustered, ranked and realised into grammatical English text using the Simple NLG generator[1]. In (Liu, et al. 2015), an AMR (Banarescu SDL, et al. 2013) semantic parser is used to create a graph-based representation of a document that is assigned weights predicted using supervised learning methods and which indicate relevance towards inclusion in a summary. Discourse parsing has also been applied to the task of creating abstracts from user reviews. (Gerani, et al. 2014) and (Gerani, et al. 2015), an RST (Mann and Thompson 1987) discourse parser and sentiment analysis methods are used to build and prune discourse graphs, which are rendered as text using a rule and template-based generation approach.

2.3 Approaches based on shallow and surface syntactic analysis

Tools supporting deep analysis may lack robustness, support for additional languages or suffer of poor performance. In addition, the intermediate representations they produce require generation methods and tools to be cast as a text. To overcome these obstacles, researchers have experimented with intermediate representation built using more shallow linguistic analysis.

Word and dependency graphs are typical representations obtained via shallow analysis where words in the source texts are used to create a graph where multiple occurrences of a

[1] https://github.com/simplenlg/simplenlg

word are merged into a single node and edges are added either between words found consecutively in the text -as in word graphs, e.g. (Filippova and Altun 2013), (Thadani and Mckeown 2013), (Mehdad, Carenini and Ng 2014), (Banerjee, Mitra and Sugiyama 2015)- or being connected through a dependency relation – as is the case in dependency graphs, e.g. (Barzilay and McKeown 2005), (Filippova and Strube 2008), (Elsner and Santhanam 2011), (Cheung and Penn 2014). After weighting the nodes in the graph according to metrics measuring potential relevance, generation is addressed by extracting paths, which are often ranked using language models or linguistic constraints to ensure grammaticality. While the syntactic information in dependency graphs can be used to improve grammaticality, the resulting paths must be linearised into the order in which they will be presented in the summary.

Another line of research focuses on drawing from advances in Open Information Extraction (OIE) and pattern learning for large-scale relation extraction, e.g. (Alfonseca and Pighin 2013), (Pighin, et al. 2014), (Li, Cai and Huang 2015), (Wang, et al. 2016). Text-based patterns are collected or learnt from general corpora and then matched in the text to be summarised to extract text-based tuples indicating relations between text fragments. These patterns usually contain placeholders based on grammatical categories, syntactic relations, named entity (NE) types and lexical cues. Each match of a pattern obtained by replacing the placeholders with fragments of the input text can then be used as a sentence in the summary.

2.4 Sequence to sequence models for summarisation

Sequence to sequence models originally developed for machine translation are being applied to extractive and abstractive summarisation, e.g. (Rush, Chopra and Weston 2015), (Nallapati, et al. 2016), (Takase, et al. 2016), (See, Liu and Manning 2017), (Paulus, Xiong and Socher 2018). The main advantage of these models is that they do not require any intermediate representation or text generation mechanism. Grammaticality is typically guaranteed by incorporating language models into the network. The models used for summarisation are modified to extract words from the input under certain circumstances to be able to produce rare or unseen words, and to consider the words already generated to avoid repetitions.

Crucial to the success of such systems is the creation of large summarisation datasets such as the CNN/Daily Mail2 and the New York Times (NYT) (Sandhaus 2008). While neural systems excel when applied to the production of one-sentence or few-sentences summaries and evaluated using overlap metrics such as those in the ROUGE package3 (Lin 2004), the overall coherence and semantic soundness of longer summaries resulting from paraphrasing the input text may be lacking.

[2] https://cs.nyu.edu/~kcho/DMQA/

[3] https://rxnlp.com/how-rouge-works-for-evaluation-of-summarization-tasks/#.XKyf29v7RhE

2.5 Multilingual text generation

For targeting the development of a reusable Natural Language Generation (NLG) pipeline and its interface with the Knowledge Base (KB), 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 y Korelsky 1992). In V4Design, step (i) is carried out by the Reasoning module and/or the content selection module described in this deliverable, and steps (ii) and (iii) by the text generation module.

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 in 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 (Gardent, Shimorina, et al. 2017). The main problems with these approaches are their low portability to new languages and domains and the lack of control over the final output, but also the very limited amount of actual linguistic knowledge used during the generation process. A multilayer 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). The only grammar-based system used successfully in all recent NLG shared tasks is FORGe, the open-source generator developed in the framework of V4Design, which addresses the last two NLG subtasks mentioned above, namely text planning and linguistic generation. FORGe, building on the lines of the Meaning-Text Theory (Mel'čuk 1988), is based on the notion of linguistic dependencies, that is, the semantic, syntactic and morphologic relations between the components of the sentence. It was the best system at the WebNLG 2017 shared task (automatic verbalisation in English of several hundreds of pre-selected properties) according to all human evaluations, and was the most portable generator, with the best results for all metrics on unseen data.[4] FORGe is a very promising system, but currently handles only a small subset of abstract contents, its text planning layer is embryonic, and its linguistic generation layers suffer from coverage issues, due to the fact that this generator has been developed in the framework of EU projects that always target specific domains (Wanner, et al. 2010), (Bouayad-Agha, et al. 2012), (Wanner, et al. 2015).[5] On complex general-domain inputs, for about 25% of the contents, the generator does not find an adequate syntactic structure and cannot generate complete sentences; furthermore, its multilingual coverage is limited (Mille, et al. 2017). Thus, one of the main objectives of V4Design with respect to text generation is to improve the multilingual coverage and the quality of the UPF generator.

[4] http://webnlg.loria.fr/pages/webnlg-human-evaluation-results.pdf
[5] See, e.g., the FP7 and H2020 projects PESCaDO, and KRISTINA.

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 on 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 to ensure reusability (e.g. VerbNet (Schuler 2005), NomBank (Meyers, et al. 2004) and PropBank (Kingsbury and Palmer 2002), which, thanks to the amount of multilingual resources connected to them, can be used as interlingua). 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 (SW) approaches such as Lemon (Walter, Unger and Cimiano 2014) or word embeddings (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 SW 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-friendly.

3 BASIC TECHNIQUES FOR SUMMARISATION

The summarisation task, defined as T5.4 in the DoA, consists in applying text planning techniques, that is, in choosing from the contents available to the KB and structuring those contents in a way that ensures that an informative, non-redundant, coherent and user-oriented summary can be produced using the multilingual text generation methods foreseen in T5.5. In this section we will describe the basic techniques for text planning developed during the first half the project and some of the work towards more elaborate methods. These basic techniques cover the production of extractive summaries, i.e. summaries composed of fragments of the source text, and are largely based on the results of a linguistic analysis. In opposition, advanced techniques aim at incorporating ontological representations to produce a plan of an abstractive summary, one that does not replicate parts of the original text but is generated anew using NLG methods.

The basic version of text planning is based on the idea of ranking text fragments. Unlike other ranking-based approaches to extractive summarisation we base our ranking on the results of a deep linguistic analysis of text that includes both deep syntactic parsing and WSD/EL. The ranking methods presented here reuse some ideas we applied for text planning and disambiguation tasks in previous projects, i.e. MULTISENSOR[6] and beAWARE[7]. Our text planning method for MULTISENSOR (Mille, et al. 2016) consisted in ranking BabelNet synsets produced by Babelfy[8], a WSD/EL tool based on BabelNet[9]. The ranking was driven by a simple frequency metric based on the co-occurrence of pairs of synsets in a sense-annotated corpus. We elaborate on this approach by incorporating a new biased ranking method capable of incorporating more elaborate criteria based on text and sense embeddings, and jointly addressing WSD/EL and the salience ranking for text planning.

In beAWARE we address WSD/EL against BabelNet using the same bias ranking method and criteria based on embeddings as in V4Design. For this project, however, we extend the method to produce a ranking not just of senses but also of text fragments, thus enabling the production of both extractive and abstractive summaries -the former by selecting parts of the source text based on their ranks, the latter by applying advanced text planning and linguistic realisation methods.

3.1 Input representation of contents

As mentioned in the DoA, we foresee a hybrid approach for text planning where the system draws, on one hand, from the results of the linguistic analysis of input texts in WP3 (T3.2 and T3.3) and, on the other, from the ontological representations and linked data resulting from the integration and reasoning tasks in WP5.

[6] https://www.multisensorproject.eu/

[7] https://beaware-project.eu/

[8] http://babelfy.org/

[9] https://babelnet.org/

The basic techniques presented here focus on the linguistic representations produced in T3.2 to T3.3 as result of WSD, EL and dependency-based deep parsing. Consequently, the input to text planning for summarisation is defined in terms of deep syntactic or predicate-argument relations holding between pairs of references to lexical and knowledge resources such as DBpedia[10], BabelNet[11], PropBank[12], NomBank[13], VerbNet[14] and FrameNet[15].

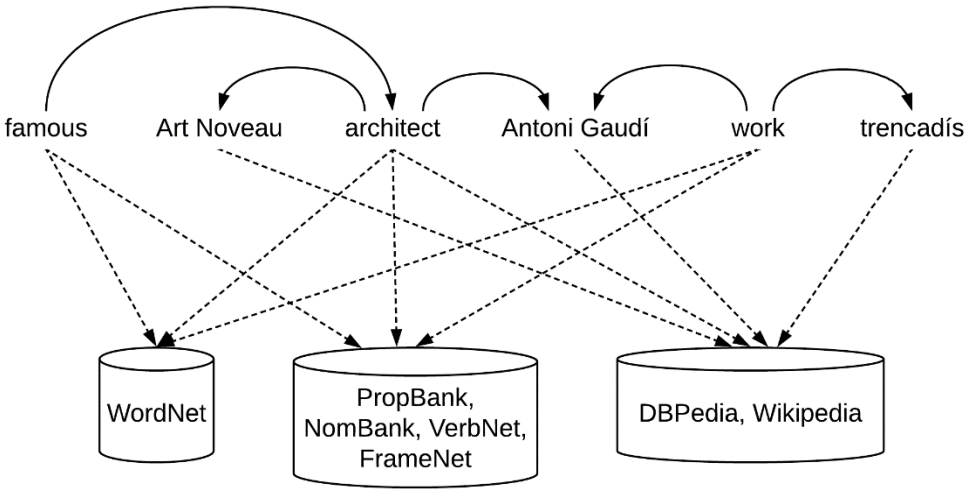


Figure 2: Example of input contents

The linguistic representation resulting from analysing the sentence "Antoni Gaudí, the famous Art Nouveau architect, introduced *trencadís*" is illustrated in Figure 2. NEs such as "Art Nouveau" and "Antoni Gaudí" are detected and linked to their correct entries in DBpedia, Wikipedia and BabelNet by addressing named entity recognition (NER) and EL tasks. Concepts like "famous", "architect", "work" and "trencadís" are also disambiguated to the correct entries in the aforementioned resources and WordNet by applying WSD. Finally, deep dependency-based parsing is responsible for finding the correct senses for predicative words like "famous", "architect" and "work" in PropBank, NomBank, VerbNet and FrameNet. In addition, parsing also establishes the participants of each predicate and their roles -as indicated by the arrows on the upper part of the figure. Deep parsing will be described in detail in the corresponding deliverables for WP3 (D3.3, M18). While WSD and EL also belong to WP3, we address them jointly with text planning tasks of WP5 and are therefore described in this deliverable.

[10] https://wiki.dbpedia.org/

[11] https://babelnet.org/

[12] https://propbank.github.io/

[13] https://nlp.cs.nyu.edu/meyers/NomBank.html

[14] https://verbs.colorado.edu/~mpalmer/projects/verbnet.html

[15] https://framenet.icsi.berkeley.edu/fndrupal/

3.2 Ranking and disambiguation of meanings

The first step in the text planning task is to rank the potential meanings -named entities or concepts- mentioned in the source text. This ranking jointly addresses WSD/EL (T3.1) and the determination of salience required for T5.4. We treat the ranking of potential meanings as a centrality problem, where the most *central* meanings correspond to the most salient. We adopt as criteria to estimate salience a similarity metric between pairs of meanings and a context-based metric for single meanings.

Similarity of meanings is estimated using sense embeddings, i.e. vectors indicating distributional properties of meanings learnt from corpora. More precisely, the similarity of two meanings is calculated as the cosine angle between their vectors. Our current version of the ranking component employs SEW-Embed vectors (Delli Bovi i Raganato 2018), but additional embeddings may be considered in the future.

Our context-based metric aims to filter out candidate synsets that are found to be unlikely candidates based on the context of their mentions in the text, therefore reducing the scale of the ranking problem and time required to solve it, and to introduce an initial bias towards most likely candidates. The bias of each candidate meaning is calculated by estimating its plausibility from the average similarity between the local contexts of its mentions in the text and a set of glosses obtained from BabelNet. By introducing this context-based dimension, we effectively combine a global, similarity-based ranking approach with a local, context-based one.

Given a candidate synset s, its set of mentions M in the source text, its set of glosses G in BabelNet, and a set of word vectors W, we estimate the bias as the cosine angle between a Bag-of-Words (BoW) average of W for M and the BoW average of W for G. Calculating text vectors as the arithmetic mean of the words in the text is a simple but frequently used method applied to many tasks assessing similarity between texts (Perone, Silveira i Paula 2018). We may adopt more elaborate approaches in the future, e.g. Smooth Inverse Frequency (SIF) (Arora, Liang i Ma 2017). Our BoW only filters out function and stop words, the former detected by the deep syntactic parser and the latter using a fixed list of stop words. As word embeddings, we use pre-trained fastText vectors (Bojanowski, et al. 2017) without the model for estimating out-of-vocabulary (OOV) words.

For the ranking of meanings, we adopt the eigenvector formulation used in the Biased LexRank extractive system (Otterbacher, Erkan and Radev 2009). Given a bias vector L for each candidate meaning, a similarity matrix X defined for each pair of candidate meanings, and a damping factor d, the calculation of the biased eigenvector centrality score is described by the equation in Equation 1.

$$\boldsymbol{MR}_u = d \cdot \boldsymbol{L}_u + (1-d) \cdot \sum_{(u,v) \in E} \boldsymbol{X}_{u,v} \cdot \boldsymbol{MR}_v$$

Equation 1: biased centrality score used for the ranking tasks

In the equation above, u and v are candidate meanings, bias L_u corresponds to the cosine distance between the BoW vectors of the contexts and glosses of u, and X_{uv}corresponds to

the pair-wise similarity calculated as the cosine distance between the sense vectors of u and v.

Before ranking, we constrain the set of candidate meanings per mention to the top k candidates with the higher candidate scores. After the ranking, we choose the highest ranked candidate, provided its rank is above a threshold t. If no candidates are above the threshold, then we pick the first sense returned by the BabelNet API, a strategy that amounts to using the BabelNet First Sense (BFS) baseline described in (Moro i Navigli 2015) as a fall-back for our own ranking procedure -BFS prefers most frequent senses according to WordNet. The values for k and t have been set experimentally to 3 and 0.8. In the case where two or more mentions with candidate senses overlap, we assign a meaning to the mention with the highest ranked candidate and discard the meaning for the other mentions. For instance, synsets may be found in BabelNet for “renewable energies”, “renewable” and “energies”. If the highest ranked synset corresponds to “renewable energies”, then we do not assign meanings to “renewable”, nor to “energies”.

3.3 Ranking of mentions

Following the ranking of candidate meanings, the highest ranked meaning is picked for each individual mention and chosen as the correct -disambiguated- meaning for that mention. Since different mentions to the same meaning may have different salience, we estimate their individual salience by applying again the same biased ranking method used for the ranking of meanings, but this time the similarity is based on the dependency-based semantic relations connecting mentions in the analysis of each sentence of the source text, while the bias corresponds to the rank values obtained on the previous step.

This ranking at sentence level also includes words and mentions to meanings excluded from the previous ranking, either words with no candidate meanings or mentions of meanings not found in the SEW-Embed embeddings. The resulting ranking assigns higher prominence to sentence fragments with a high average meaning ranking. These results will be used to highlight salient fragments of the text and to produce extractive summaries.

3.4 First evaluation

In the following, we describe and present the results of the first evaluation of the basic techniques already developed for the V4Design project. At the point of writing this report, we have achieved conclusive results for the first pass of the ranking described in Section 3.2. While we also describe the evaluation methodology for the second ranking step in Section 3.3, we will report results in the next WP5 prototype. We have conducted two separate evaluations on the ranking of candidate meanings, one designed to assess its performance in the WSD and EL tasks, and another designed to evaluate the salience of meanings estimated by the computed rank values.

Our first evaluation uses the English version of the multilingual dataset created for SemEval-2015 Task 13 (Moro i Navigli 2015), which contains four documents with 1426 tokens and annotated with 1261 references to synsets in BabelNet 2.5.1[16]. Nine participating systems

[16] Our System used BabelNet 3.7, which may lead to some results being evaluated as false positives if the chosen meaning is in 3.7 but not in 2.5.1.

and a BabelNet first sense (BFS) baseline were evaluated using precision, recall and F1 metrics.

Table 1: Results of WSD and EL evaluation

System	All	NEs	Word senses				
			All	N	V	R	A
TALN-WSD	**70.1**	88.5	**68.7**	**71.4**	55.1	73.2	**83.2**
LIMSI	65.8	82.9	64.7	64.8	56.0	76.5	79.5
SUDOKU-Run2	61.6	87.0	59.9	62.5	49.6	70.4	71.7
DFKI	-	**88.9**	-	70.3	**57.7**	-	-
BFS	67.5	85.7	66.3	66.7	55.1	**82.1**	82.5

Table 1 shows the F1 scores of the two participating systems with best overall score (LIMSI, SUDOKU-Run2), the participating system with best score of NEs (DFKI), the BFS baseline and our system (TALN-WSD). As seen in the table, none of the participants managed to beat the baseline for English for all annotations, and only two systems achieved higher F1 for NEs. Our system manages to beat the baseline in the overall score (70.1). We believe that using the BFS strategy as a fallback in our implementation helps our system to achieve good results: in our evaluation, 71% of the predicted meanings were decided by falling back to the BFS baseline instead of using the ranking values. While our system is slightly behind the best performing system in disambiguating NEs, it outperforms both the baseline and all participants in disambiguating nouns (N) and adjectives (A). Lower performance with verbs (V) and adverbs (R) is probably due to the use of the whole document as a context when calculating the bias. These scores could be improved using a local window of words as context coupled with supervised learning methods.

Our next evaluation assesses the results of the ranking of meanings. We use a set of five documents randomly drawn from the DeepMind QA dataset (Hermann, et al. 2015) -a corpus consisting of news articles from the Daily Mai and CNN, each article accompanied with human-written abstractive summaries 3-4 sentences long[17]. The five summaries have been manually annotated with synsets in BabelNet matching the meanings communicated in the text. We allow more than one synset per mention whenever multiple synsets are judged correct. We have chosen Mean Average Precision (MAP) as our evaluation metric for the ranking, with a relevance function rel that checks if the synset at position k is annotated in the summary. In other words, given a document D and the set of gold meanings G annotated in the summary of D, then $rel(k) = true$ iff $k \in G$. This metric rewards our algorithm for placing a high number of gold meanings in the first positions of the ranking, while it does not penalise our system for adding additional meanings to the ranking – this suits us because we want to produce a complete ranking of the candidate meanings that can be used to disambiguate all mentions, even those where none of the candidates are at the top of the list.

[17] Future versions of text planning incorporating the project ontologies will be conducted on texts and summaries pertinent to the V4Design use cases.

Table 2: Results of the evaluation of meanings

	All	Multiwords	N	V	R	A
TALN-UPF	**0.31**	**0.38**	**0.35**	**0.28**	**0.3**	**0.34**
Baseline	0.28	0.36	0.29	0.21	0.18	0.30

As a baseline we use BFS to disambiguate mentions in the text and automatically prefer longer mentions over shorter overlapping ones -e.g. if “renewable energies” is indexed in BabelNet, we choose a synset for it using BFS and do not choose synsets for “renewable” nor “energies”. The ranking is then produced from the disambiguated senses by ranking them by number of mentions to the synset and then by position in the source text of its first mention. Table 2 shows the MAP values for the baseline and our system. Unfortunately, we cannot compare to other systems because, as far as we know, no existing summarisation nor text planning systems implement this type of ranking. While our system manages to beat the baseline, we expect future versions to improve the gain MAP. Future evaluations on the downstream application of this ranking to summarisation should also shed light on its performance and allow us to compare with other approaches to summarisation.

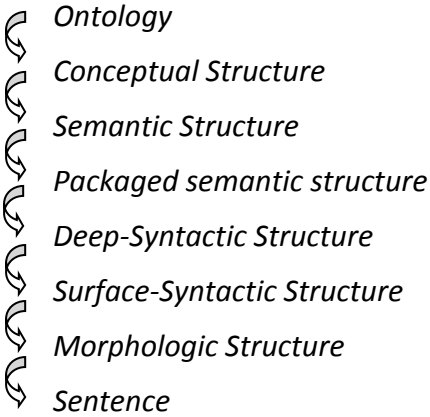
4 BASIC TECHNIQUES FOR MULTILINGUAL TEXT GENERATION

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 description generator that is coordinated with V4Design's Knowledge Base (T5.1, T5.2). The work has been carried out as foreseen during the first 16 months; thus, the present deliverable contains a description of the approach (Section 4.1) and the respective implementation (Section 4.9). This section also contains a part that summarises what has been advanced on during the first year and a half of V4Design, which includes a report on preliminary quantitative evaluations performed before Prototype 1 (Section 4.10).

4.1 Approach

In our approach, as mentioned in Section 2.5 , the text generation consists of two sub-modules: sentence packaging (aka text planning) and linguistic generation; the latter is in its turn split into several modules that address the tasks of sentence structuring (choosing the words to be used and organise 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 (T5.1, T5.2), and the ontological structures must thus be mapped to linguistic structures before the process can start. The generation is performed step by step, by successively mapping one level of representation onto the adjacent one:



In the next subsections, we describe the role of each transition.

4.2 From Ontology to Conceptual Structure (ConS): making the structure linguistically motivated

The mapping of the ontological representation to a conceptual one is the first step towards the projection of the input class and property assertions to language-oriented structures. The ontological-to-conceptual grounding is based on the Description and Situation (DnS)

pattern of DOLCE+DnS Ultralite[18] (dul). Under the adopted DnS-based paradigm, each dul:Situation object corresponds to an n-ary linguistic predicate, with its participating entities' roles specified through the associated dul:Concept objects that are in turn defined (via dul:satisfied assertions) by corresponding dul:Description objects. When mapping to the conceptual structure, participating elements classified as arguments are mapped to linguistic arguments (i.e. labelled edges that link the predicate to the argument), while circumstantials (e.g. temporal attributes of an action, such as start time and duration) are treated as typed predicative nodes, through which the linking between the predicate and the circumstantial entity is realised. Thereby, as opposed to the content 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. Some other elements are removed altogether (e.g. the situation and description elements), while others are captured in another form. For instance, the information related to the ontological type of retrieved data, will only be realised as grammatical tense on the main verb of the sentences (in the case of habits, present), and not mentioned as such. Table 3 summarises the main transformation rules used for mapping the DnS-based representations to respective conceptual structures.

Table 3: DnS-based to ConS-based representation transformation mappings.

DnS-based representation	***Conceptual structure representation***
Eventive relational contexts, i.e. dul:Situations (onto:Sleep, onto:Eat, onto:Walk, etc.) and participating entities	N-ary predicates and their arguments
Argumentative dul:Concept specialisations (context:Agent, context:Beneficiary, context:Theme, context:Destination, etc.)	Argument labels of referenced n-ary predicates (Argument1, Argument2, etc.)
Circumstantial dul:Concept specialisations (context:FrequencyAttribute, context:TemporalAttribute, context:TemporalPattern, etc.)	Typed predicative nodes (Frequency, StateTime/EndTime, TemporalOverlap/ TemporalOrder, etc.)

Grounding the conceptual representation on the DnS model and retaining a high level of abstraction allows for a principle interface between the ontology and the semantic structures of the next layers while rendering it sufficiently generic, so as to allow for generation in any language. At this point in the project, the conceptual structures are in the form of simple predicate-argument templates associated to each property in the ontology. As an example, consider Figure 3 that illustrates a triple set as delivered from the KB, which describes the building called "250 Delaware Avenue", and its respective "conceptual" representation, illustrated in Figure 4 as a set of populated predicate-argument templates.

style (250 Delaware Avenue, postmodern_architecture)

floor_area (250 Delaware Avenue, 30,843.8)

floor (250 Delaware Avenue, 12)

construction (250 Delaware Avenue, January 2014)

[18] http://www.ontologydesignpatterns.org/ont/dul/DUL.owl

location (250 Delaware Avenue, Buffalo (New York))

Figure 3: Five triples related to the 250 Delaware Avenue building

Note that the conceptual structures are linguistic structures, since they contain only linguistic elements (meanings, or concepts). However, they are language-independent, in that the same input structure is used whatever the target language is. Multilinguality is dealt with during the next transition.

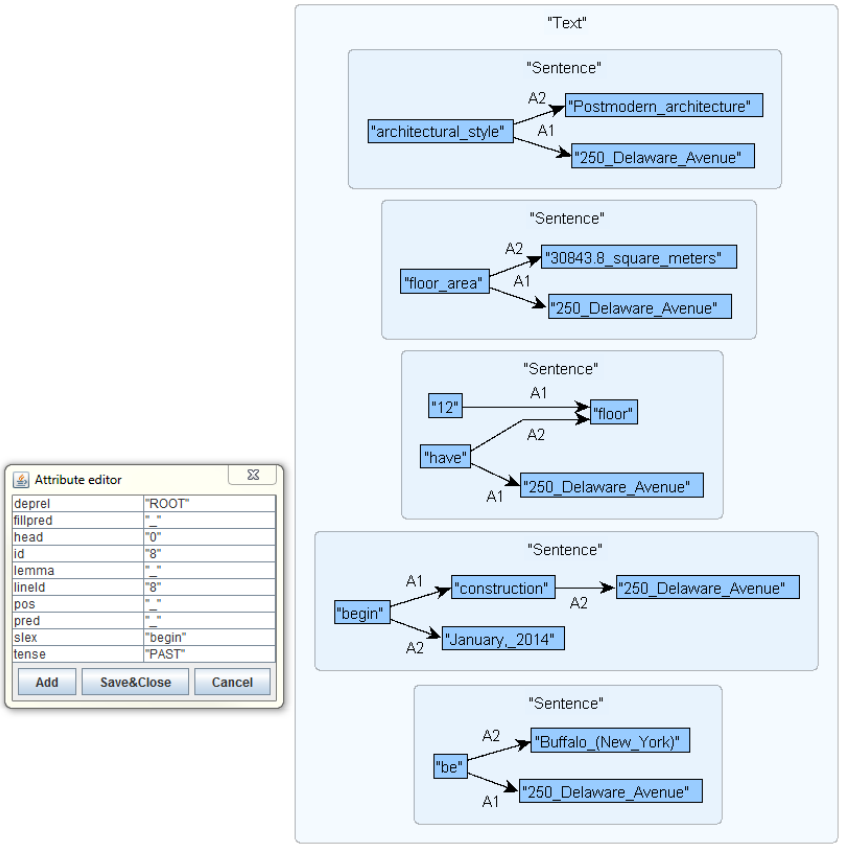


Figure 4: Five populated predicate argument structures corresponding to the triples in Figure 3.

4.3 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, Spanish, Greek, and German). In order to illustrate the difference between a concept and a semanteme, consider the case of the concept of *measurement*. For example, the wind, as a physical event, can be measured, and this can be expressed in combination with the meaning *speed* in English. By contrast, in some languages, no meaning is available in order to realise *speed* in combination with *wind*. Mentioning *wind* with a rating is enough in order to understand that we are talking about wind speed. In English too it is actually common not to mention *speed*, and the organisation of the concepts must allow for choosing one way or another of combining the meanings. 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 V4Design, a simplified and more practical view has been applied, considering that 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 the 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.

4.4 Text planning / Sentence packaging: defining the boundaries of sentences

If several discursive units such as the one shown in Figure 4 (the group of nodes called "Sentence" is what we call here a discursive unit) are fed to the generator, they will each be realised by default as independent sentences. 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 argument in the input: if the object arguments of two unlinked predicates have the same relation with their respective predicates, they will be coordinated. For instance, if the generator receives two separate units corresponding to *Mr X built building Y* and *Mr X built building Z,* these two units will be rendered as one single sentence *Mr X built buildings Y and Z.*

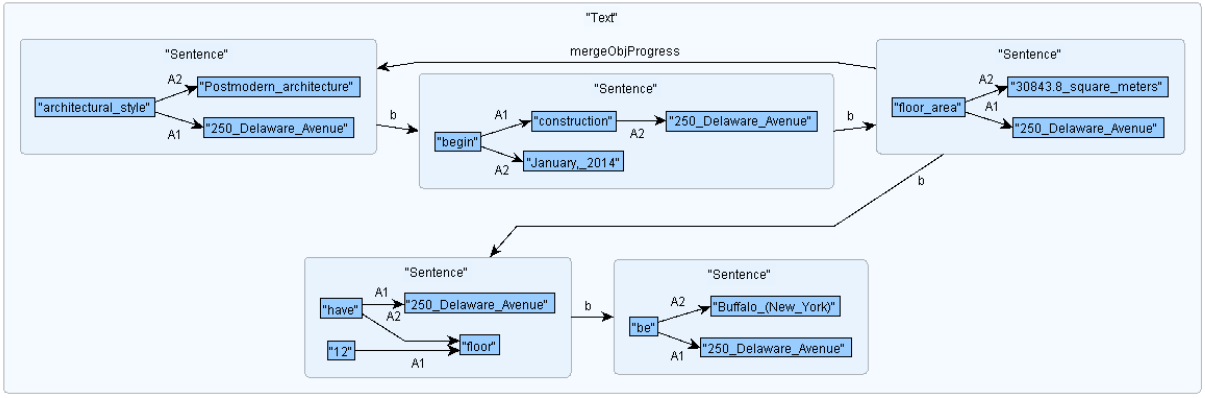


Figure 5: Sentence packaging through aggregation of triples

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. For instance, in our example in Figure 5, the first and third triples have the same argument '250 Delaware Avenue' with different main predicates ('architectural_style' and 'floor_area') and relations (A2 and A1 respectively): the third triple will be merged with the first one (as shown in Figure 6) and the final sentence will be something like The style of 250 Delaware Avenue, which has a floor area of 30,843.8 sq.m., is postmodern architecture. 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.

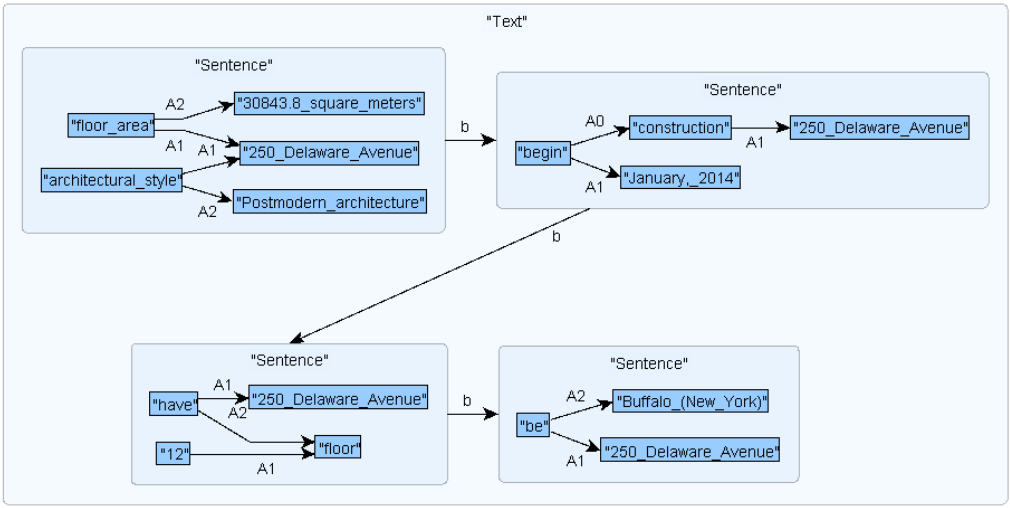


Figure 6: Semantic structures after sentence packaging.

One more action is performed during this step, which is the determination of the communicative structure. Indeed, 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 doesn't 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. In Figure 6, the nodes 'floor_area', 'begin', 'have' and 'be' have been identified as being the main node of their respective sentence. All the nodes have been assigned a part of speech and linked to an entry in a lexical resource: for instance, *begin* is linked to the entry 'begin_VB_01' according to the PropBank nomenclature (Kingsbury and Palmer 2002).

4.5 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 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 this root, the whole tree is built node after node.

A lexicon indicates what a syntactic predicate requires in order to form a correct sentence in a language (syntactic combinatorial); for instance, the verb 'begin', 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 (LFs). Those LFs 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'.

Figure 7 shows that the main syntactic node 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*). Coreferring 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.

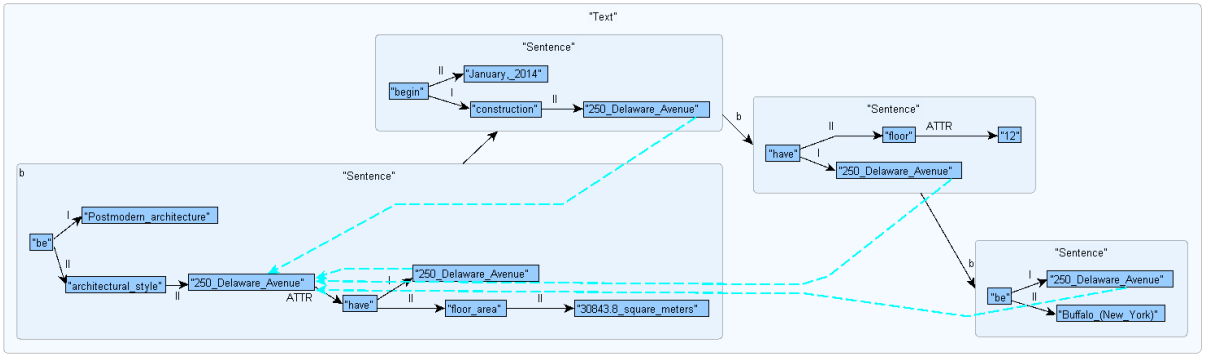


Figure 7: Deep-syntactic structures that correspond to the semantic structures in Figure 6.

4.6 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, if there is the DSynt configuration *begin-II->January 2014* i.e. the verb 'begin' with 'January 2014' as its second (II) argument, as in Figure 7, the entry of 'begin_VB_01' indicates that the dependent II must be introduced by the preposition 'in', which is then a so-called governed (bound) preposition. Other non-lexical nodes are introduced, such as governed conjunctions, auxiliaries, determiners, expletive subjects, etc.

Lexical Functions must also be resolved during this transition: most words of the lexicon are the keyword of one or more LFs. The value(s) of the LFs is stored in the entry of a word: 'heavy' as the value of the LF *Magn* in the entry of 'rain', for instance. 'Pouring' would be another value for the same LF of the same word.

Finally, the generic syntactic relations found in DSynt are refined into more idiosyncratic relations which convey very accurate syntactic information, instead of semantic, as is the case with the argument numbers. For instance, the DSynt 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. A *NMOD* cannot be promoted or demoted, does not trigger any agreement and always has to be realised to the right of its governor.

Figure 8 shows the surface-syntactic structure with functional words and language-specific syntactic relations.

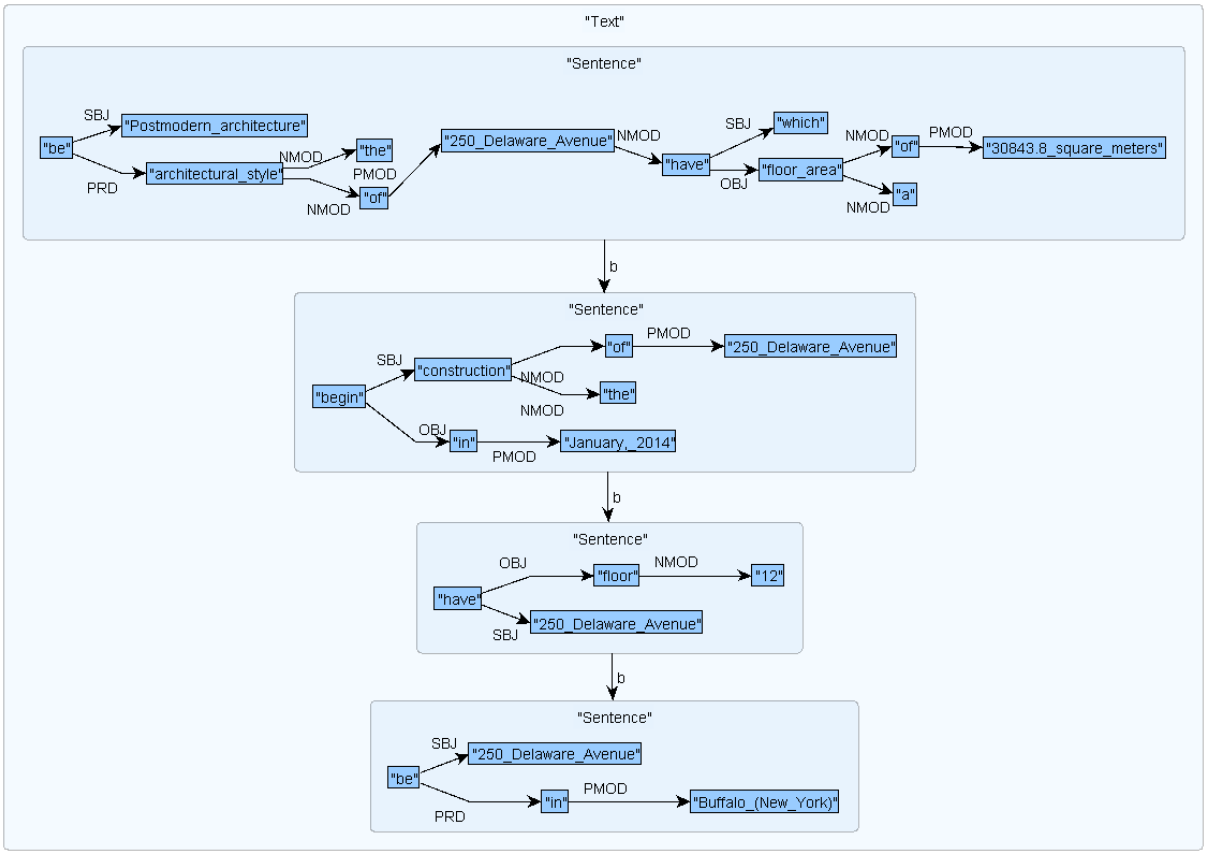


Figure 8: Surface-syntactic structures that correspond to the deep-syntactic structures in Figure 7.

4.7 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., '*begin<VB><IND><PRES><3><SG> = begins*', or by using some automata based on inflection schemas, such as Two Level Morphology, 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 precedence relations are shown in red.

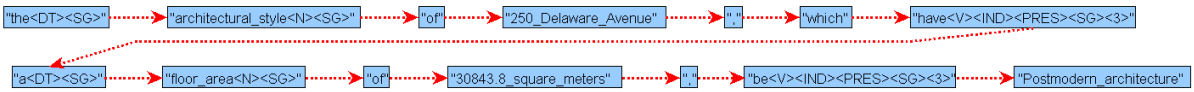


Figure 9: A (linearised) morphological structure that corresponds to the first surface-syntactic structure in Figure 8.

4.8 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:

The architectural style of 250 Delaware Avenue, which has a floor area of 30843.8 square meters, is Postmodern architecture. The construction of 250 Delaware Avenue began in January 2014. 250 Delaware Avenue has 12 floors, and it is in Buffalo (New York).

4.9 Implementation with the FORGe generator

In this section, the numbers within parentheses refer to the different mappings sketched in the previous section. We describe the basic methodology followed for the deep and surface generation processes, and report on the development of a new environment for the implementation of the rule-based generator.

Deep Generation

For the deeper stages of generation, i.e. for going from the ontology to the deep-syntactic structures used for surface generation, rule-based modules have been developed. This includes the DnS-based translation rules for mapping the ontology representations to conceptual structures for step 4.1.1 and graph-transduction grammars for steps 4.1.2, 4.1.3, 4.1.4 (UPF FORGe generator). In combination with the graph-transduction rules, a semantic dictionary, that includes equivalences between the entities in the semantic repository and the words of the different languages of the project, is required.

The DnS-based translation rules are manually crafted and realise the inverse transformation of the one applied during the analysis of the verbal content of user utterances. For the implementation of the ontology to conceptual structure translation the Jena API[19] for RDF is used for parsing the input DnS-based responses.

Surface generation (FORGe generator)

For generation from deep-syntactic structures (4.1.5, 4.1.6, 4.1.7), two different approaches will be implemented, both based on a pipeline of graph transducers, which convert the output of the text planning stage into a well-formed text, and both being implemented as

[19] https://jena.apache.org

part of the MATE tools[20]. In this section, we briefly describe the basic techniques for text generation (UPF FORGe generator); the advanced techniques are described in Section 5.7.

The basic text generation implementation consists of manually crafted graph-transduction grammars for each transition between two consecutive layers; see Table 4 for more details on the rules. 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). We manually crafted language-specific dictionaries that cover the texts foreseen for the first prototype.

In order to reach large-coverage, we have been experimenting on the extraction of subcategorisation patterns from lexico-semantic resources such as PropBank (Kingsbury and Palmer, 2002), NomBank (Meyers et al., 2004) and VerbNet (Schuler, 2005); see (Mille and Wanner, 2015). The following sample entry shows the syntactic properties of the verb 'give', which has three nominal arguments, the third one being introduced by the preposition 'to' (*give something to someone*):

```
"give_VB_01":_verbExtArg_{
vncls = "13.1"
pbID = "give.01"
pbsenseID = "01"
   lemma = "give"
gp = {
    I = {
      pos = NN
      rel = SBJ
    }
    II = {
      pos = NN
      rel = OBJ
    }
    III = {
      pos = NN
      rel = IOBJ
      prep = "to"
    }
  }
}
```

The method has proven to be useful in English, but not successful enough to be applied to other languages so far, especially given the lack of parallel resources in PropBank and NomBank for these languages. The generation module covers all languages involved in interactions with the Knowledge Base in the project, that is, English, German, Greek, and Spanish, with different coverages, directly related with the size of the respective lexicons (see Table 4).

4.10 Advances in V4Design

In this section, we report on the advances made in the framework of V4Design during the first 18 months of the project.

[20] http://code.google.com/p/mate-tools/

Table 4 shows the state of the FORGe language generator at the beginning of the V4Design project (01/01/2018), as found in the final deliverable of the H2020 KRISTINA project (project number 645012, D6.3, 31/12/2017), with additional information on the English resources (not reported in KRISTINA).

Table 4: State of the V4Design's NLG system

		V4Design M0	**V4Design M18**
Languages supported		EN, ES, DE	EN, ES, DE, EL
Number of rules	ALL	1,373	1,605
% of language-independent rules		Con-SMorph (1,373) : 70%	Con-SMorph (1,605) : 75%
		Con-Sem (416) : 94% Aggregation (212) : 91% Sem-DSynt (177) : 75% DSynt-SSynt (307) : 39% SSynt-DMorph (172) : 48% DMorph-SMorph (89) : 55%	Con-Sem (459) : 98% Aggregation (216) : 100% Sem-DSynt (202) : 70% DSynt-SSynt (397) : 50% SSynt-DMorph (188) : 69% DMorph-SMorph (143) : 49%
Main linguistic phenomena supported and improvements	EN	Basic sentence planning Basic sentence structures: Argumental dependents Temporal circumstancials Other circumstancials Coordinations Embedded clauses (relatives) Complex syntactic structures Lexicon-based introduction of functional elements: Functional prepositions and conjunctions Modals Basic verbal and determiner agreements	Improved linearisation Improved aggregation strategies Basic referring expressing expression generation Basic structure well-formedness checking Complex support verb constructions Complex relative clause construction Advanced verb agreements (coordinations)
	DE	Basic sentence planning Basic sentence structures: Argumental dependents Circumstancials Coordinations Embedded clauses Complex syntactic structures Lexicon-based introduction of functional elements: Functional prepositions and conjunctions Modals and auxiliaries Verbal, adjectival and determiner agreements Nominal compositionality	
	ES	Advanced sentence planning Advanced sentence structures: Argumental dependents Circumstancials Coordinations Embedded clauses Lexicon-based introduction of functional elements: Functional prepositions and conjunctions Modals, auxiliaries	

		Verbal, adjectival and determiner agreements	
	EL		Basic sentence planning Basic sentence structures: Argumental dependents Circumstancials Coordinations Embedded clauses Lexicon-based introduction of functional elements: Functional prepositions and conjunctions Modals and auxiliaries Verbal, adjectival and determiner agreements Basic linearisation
Number of lexical units in lexicon	EN	42,694	42,712
	DE	419	457
	ES	533	1,820
	EL	0	15
BLEU score	**EN**	**36.69**	**39.84**

In the above table, a quantitative assessment of the generator is reported, first with a count of the rules and lexical entries and a description of the covered phenomena in the different languages, and then with an objective evaluation of the quality of the outputs generated in English (last row). The rule sets have generally been made more language independent; only the Sem-DSynt and DMorph-SMorph transitions contain a higher proportion of language-specific rules. For the Sem-DSynt transition, this is because some temporary rules are currently substituting information that should be in the lexicon. On the long term, these rules will be replaced by (fewer) language-independent rules and the language-specific information will be stored in the respective lexicons. The reason is different for the DMorph-SMorph transition, which is language-specific by nature, with the modelling of phenomena that are often highly idiosyncratic. Increasing the coverage of this grammar usually means adding language-specific rules, which makes their proportion increase.

For the objective evaluation, we use the BLEU metric, as foreseen in D1.2. BLEU is an n-gram-based comparison score obtained by comparing a predicted output, produced by our generator, with the expected one: single words, bigrams (sequences of two words), trigrams and quadrigrams in both outputs are compared and the similarity between them is calculated. As dataset, we use the whole evaluation section of the dependency version of the Penn Treebank (Johansson and Nugues 2007), converted to predicate-argument structures as illustrated in Figure 6, using the semantic analyser described in (Mille, et al. 2017). In order to make the results fully comparable with the results at Month 0 (Table 4, third column), the converted semantic structures are then sent to the FORGe generator, replacing the linearisation module by the same off-the-shelf linearisation tool used for the previous evaluation. As a result, the improvement of the score is due almost exclusively to the Sem-DSynt and DSynt-SSynt grammars, which are the core of the generation pipeline. The last row of Table 4 shows that the BLEU score increased 3.15 points, from 36.69 to 39.84, which represents an increase of 8.6%, better than the lower expectation and close to the highest expectation from D1.2.

In summary, five main improvements were made to the UPF generator:

- the general coverage of the rules was improved: all grammars are now more complete, with 1,605 rules in total, as opposed to 1,373 at the beginning of the project;
- thanks to the updated rules, the quality of the English generator has been significantly improved on a challenging dataset (+8.6% BLEU score);
- rules were made more language independent: now, 75% of the rules are language-independent, as opposed to 70% at the beginning of the project;
- basic support has been added for Greek, together with a toy lexicon;
- the size of the Spanish lexicon has been significantly increased, with more than three times as many entries as at the beginning of the project (1,820 entries now).

5 TOWARDS AN ADVANCED SUMMARISATION STRATEGY

In this section, we report on the advanced strategies for text generation in the context of abstractive summarisation. This includes the plans for the text planning module (Section 5.1) and the multilingual surface generator (in Section 5.5), and some results on the sentence packaging module (in Section 5.5).

5.1 Text planning

In this section, we describe the plans for the second part of the project with respect to text planning.

5.2 Incorporating ontological representations

The two-pass ranking procedure described in Section 3 produces a rank of the most salient fragments of a text without accounting for any domain- or application-specific aspects. To adapt text planning to aspects pertinent to user types, 3D modelling, architecture and video games, a mapping will be created from meanings in the corpora compiled in T3.1 to ontological classes. This mapping will form the basis to tune the bias of the ranking of mentions to specific ontological representations. In other words, it will enable us to tune the ranking procedure according to specific user profiles and use cases.

For the creation of the mapping between meanings and ontologies, both manual and automatic methods will be considered. The resulting mapping will also contribute towards the production of RDF-ready output as foreseen in T3.4. In addition, it will facilitate transferring ranking scores to the semantic repository, i.e. by taking the ranking scores assigned to mentions and assigning them as weights to named individuals in the repository. These weights can then be used to plan summaries based on ontological representations.

5.3 From extractive to abstractive summarisation

The ranking-based techniques for text planning described in Section 3 are limited to the identification of salient parts of the texts analysed by the system. To satisfy the goals set out for T5.4 and T5.5 in the DoA, our basic techniques need to be improved in several ways. First, the ranking procedure should be adaptable to each application context, i.e. specific use case, addressed user, etc. This will be addressed by biasing the ranking methods according to the knowledge present in the project's semantic repository. Thus, for instance, meanings related to video games can be given prominence in a use case involving the creation of VR models.

Secondly, we will experiment with data-driven algorithms capable of dynamically adding contents to a plan of the summary to be generated. More precisely, we will adopt a graph view on the semantic repository and, starting from highly weighted contents, incrementally add new contents that can be reached via the ontological and linked data relations in the repository. The algorithms considered for this task will have to balance informativeness of the selected contents with other concerns such as avoiding redundancy and preferring conciseness in the resulting summary.

Finally, the ordering of selected contents will be approached in a more principled way by incorporating concerns related to discourse structuring and information packaging such as global and local coherence, and theme-rheme progressions.

We also plan for possible contingencies. One such contingency is the case where the relations and links available in the semantic repository do not help in the discovery of relevant contents using data driven methods, either because of shortcomings in the discovery and inference of relations in other work packages or because of the existing links not being good indicators for the text planning task. In this case, the deep dependencies produced by the semantic parser will be considered as links between contents. This would place our abstractive summarisation approach in line with the summarisation systems based in deep linguistic analysis reviewed in Section 2.2.

Another contingency is poor performance or lack of coverage in the text generation component. In this case, we will consider addressing the production of summaries using neural paraphrasing models like those reviewed in Section 2.4, which would be modified to be guided by the ranking values and other planning criteria described above.

5.4 A more comprehensive evaluation

The evaluation procedure will also be extended and improved in various ways. First, texts pertinent to the project's use cases will be compiled and annotated to assess the performance of our mechanisms in the scenarios foreseen in V4Design. Second, we will improve the annotation procedure by incorporating multiple annotators and reporting inter-annotator agreement figures. Third, we will extend the datasets and corresponding evaluation to all the languages of V4Design. And last of all, we will conduct an overall evaluation of the final summaries produced using both extractive and abstractive methods and based on ROUGE (Lin 2004) or similar metrics.

5.5 Advanced multilingual text generation

The advanced implementation will be fully reported and evaluated in the final deliverable (D5.5). It consists in performing some transitions with statistical modules trained on annotated data. Indeed, by aligning node by node a parallel corpus of two consecutive levels of representation, it is possible to apply Machine Learning techniques and obtain models for a statistical generator. In prevision of developing these statistical modules, multilingual corpora containing linguistic annotations have been automatically annotated by UPF. In this section we report on the annotation and use of the data for the advanced sentence packaging and surface realisation.

5.6 Statistical sentence packaging

Sentence packaging is one of the main tasks to be solved towards abstractive summarisation from abstract ontological (Bouayad-Agha, et al. 2012) (Gardent, et al. 2017) or semantic (Bohnet, et al. 2010) (Flanigan, et al. 2016) structures.

Approach

A semantic graph to which the problem of sentence packaging to be applied is a labelled graph with semantemes, i.e., word sense disambiguated lexical items, as vertex labels and

predicative argument relations as edge labels, as seen in Figure 5. However, in the framework of abstractive summarisation, the input material is texts, and the semantic structures are much more complex than the ones seen in Figure 5. As a result, unlike the basic sentence packaging described in Section 4.4, which has as objective to group together triples in a single sentence, the advanced sentence packaging aims at splitting a large graph into smaller sentences.

A method developed within V4Design to address this task is based on clustering algorithms designed for the so-called community detection task. In the semantic graph, the vertice labels are assumed to be typed in terms of semantic categories such as 'action', 'object', 'property', etc. A semantic graph of this kind can be an Abstract Meaning Representation (AMR) (Banarescu SDL, et al. 2013) obtained from the fusion of coreference vertices across individual sentential AMRs or a VerbNet or FrameNet structure obtained from the merge of sentential Verb-Net respectively FrameNet structures that contain coreferences. An RDF-triple store, which is annotated with semantic metadata, e.g., in OWL[21] can be equally converted into such a graph (Rodriguez-Garcia and Hoehndorf 2018). Without loss of generality, we will assume, in what follows, that our semantic graphs are hybrid VerbNet / Framenet graphs in that we use first level VerbNet / FrameNet type ids as vertice labels and VerbNet relations as edge labels.

The generation information that characterizes a graph in the context of sentence packaging concerns: (i) the optimal number of sentences into which this given graph can be divided, and (ii) the profile (in semantic or graph theory terms) of a typical sentence of this graph. We use this information in the subsequent stages of sentence packaging.

In order to estimate the number of sentences into which a given semantic graph is to be decomposed, a linear regression model should be created based on various linguistic and statistical features incorporated in a graph, such as number of tokens, number of edges, number of predicate nodes (nodes with output edges) of each type, number of argument nodes of each type, number of nodes corresponding to some action according to VerbNet classes.

In order to obtain the prototypical profiles of the sentences, the following features that play an important role in sentence formation should be used in addition to features listed above: the type(s) of the parent node(s) of each node and the types of its arguments. With these enriched features at hand, a multivariate normal distribution (MVN) of the most common non-correlated features of sentences is to be built and used for assessing the correctness of sentence structure based on its proximity to this distribution.

A descent search consists in adding neighbour vertices one by one to each subgraph obtained by community detection clustering and keeping them if the correspondence of the subgraph to the multivariate distribution increased.

The overall algorithm for sentence packaging is as follows:

1. Predict the number of sentences the graph should be split with a pre-defined linear regression model.

[21] https://www.w3.org/OWL/

2. Apply community detection-based clustering that takes a predicted number of sentences as an input.
3. Optimise structure of sentences according to a pre-built multivariate normal distribution:
    i) For each $s \in S$, with S: = set of sentence subgraphs obtained by clustering algorithms
        (a) determine the degree of correspondence to the joint distribution (in case of several subgraphs, choose the most appropriate one) that is to be optimised;
        (b) apply local search descent, adding nodes from $s' \in S$ (with $s' \neq s$) iteratively each time when it leads to the increase of the optimised parameter (subgraphs can share common nodes, i.e., overlap);
    ii) Stop local search descent when there is no node that improves s.

Implementation

We used the VerbNet/FrameNet annotated version of the Penn TreeBank (Mille, et al. 2017) to which we applied the co-reference resolution from Stanford OpenCore NLP[22] to obtain a graph representation (and which we split into a development set and test set, with 85% and 15% texts that contained 78% and 22% of the sentences respectively).

Consider the schematic representation of the semantic graph of one of the texts from the development set in Figure 10. It consists of two isolated subgraphs: one of them (to the left) comprises three sentences and the second (to the right) corresponds to a single sentence. The blue (dark) nodes correspond to verbal and nominal predicate tokens.

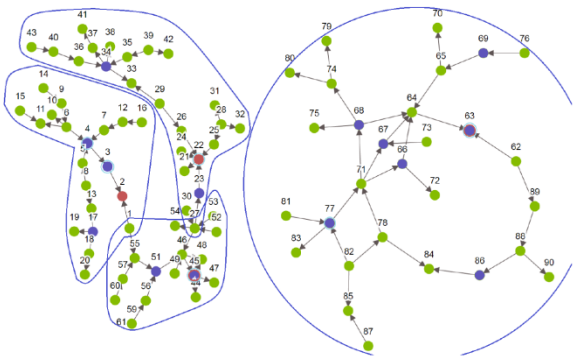


Figure 10: Example of a semantic graph of a text

A linear regression model was built using Ridge regularisation on the development set with the features listed in the first column of Table 5. The statistics on chosen features are shown in the other columns, where Q_2 is a median, *N1* is an absolute number of sentences with a non-zero value of a parameter, and *N2* is a corresponding relative number.

Table 5: Statistics of the features for the linear regression model

	min	Q_2	mean	max	N_1	N_2
# tokens	2	17	17.5	95	28253	1.0

[22] https://stanfordnlp.github.io/CoreNLP/

# edges	1	21	21.7	130	28253	1.0
# predicate nodes	0	11	11.3	67	28189	0.99
# argument nodes	1	12	12.6	67	28253	1.0
# roots	1	4	5	37	28253	1.0
# VerbNet nodes	0	3	3.2	15	26355	0.93
# Argument1	0	6	5.9	40	27794	0.98
# Argument2	0	4	4.3	30	27024	0.96
# Elaboration	0	2	2.2	19	22247	0.79
# NonCore	0	0	0.7	8	13415	0.47
# Set	0	0	1.2	26	12680	0.45

The MVN distribution for determination of the prototypical sentence was built by applying Principal Component Analysis (PCA) (Joliffe 1986) to a space of the most common 100 features and selecting principal vectors that describe 90% of the variance. This procedure made the matrix of values of sentence features to be invertible, as required for the MVN distribution. Due to several variations of possible linguistic structures of a sentence, a clustering algorithm in a space of selected features (K-means, k=10) was implemented and the distributions were built separately for each cluster.

The LOUVAIN (Blondel, et al. 2008) and METIS (Karypis and Kumar 2000) community detection algorithms were chosen for basic graph split.

Evaluation

The F1-score was chosen as a measure for the comparison of the quality of decompositions obtained by different algorithms on the test set. It is calculated for each original sentence since we consider a sentence as a separate unit. Its value takes into account which part of the original sentence was covered by the obtained subgraph and how many nodes that did not belong to the original sentence were mistakenly appended. Each isolated subgraph corresponds to one unit only, although it can include several original sentences. For those original sentences that are not captured in the majority of their nodes in any individual subgraph, F1-score is equal to 0. The macro-F1, i.e. the average F1-score over all sentences, is a final measure.

The results are displayed in Table 6. 'No decomposition' refers to the case when any graph in the test set is considered to be a sentence (it can be considered as an additional baseline); '$METIS_{LR}$' stands for "METIS with linear regression as a preprocessing stage", 'DC_K' for "descent search with K-means", and '$DC_{\neg K}$' for "descent search without K-means".

Table 6: Results of testing the obtained models

	Recall	**Precision**	**F_1-score**
No decomposition	0.313	0.264	0.274
LOUVAIN	0.69	0.726	0.68
$METIS_{LR}$	0.693	0.814	0.727

LOUVAIN+DC_K	0.707	0.709	0.681
LOUVAIN+$DC_{\neg K}$	0.705	0.704	0.678
LOUVAIN+PCA+DC_K	0.701	0.714	0.681
$METIS_{LR}$+DC_K	0.73	0.792	0.738
$METIS_{LR}$+$DC_{\neg K}$	0.731	0.788	0.736
$METIS_{LR}$+PCA+DC_K	0.714	0.795	0.731

It can be observed that the local search descent with the chosen optimisation function leads to an increase of the mean F_1-score in each case. The use of a larger number of features with PCA leads to slightly poorer results, but still shows an improvement in comparison to the baseline community detection (LOUVAIN, and $METIS_{LR}$). However, $METIS_{LR}$ is somewhat better than our optimisations with respect to precision and $METIS_{LR}$+$DC_{\neg K}$ is the best (even if by only a very minor margin, compared to the best F_1-score reaching $METIS_{LR}$+DC_K).

The very low figures for 'No Decomposition', i.e., the interpretation of each single graph as a sentence, show us that the problem of sentence packaging (or, in other words, decomposition of textual semantic graphs into sentential subgraphs) is indeed a relevant problem in large scale semantics-to-text generation.

For illustration, consider in Figure 11 a subgraph obtained from a larger initial graph. The original sentence that corresponds to the subgraph in Figure 11 is "*He said the company is experimenting with the technique on alfalfa, and plans to include cotton and corn, among other crops*". The subgraph corresponds to the ground truth subgraph with a precision of 0.938 and a recall of 0.882. It might be seen that the obtained subgraph contains enough information to generate a sentence with a similar meaning as the original one.

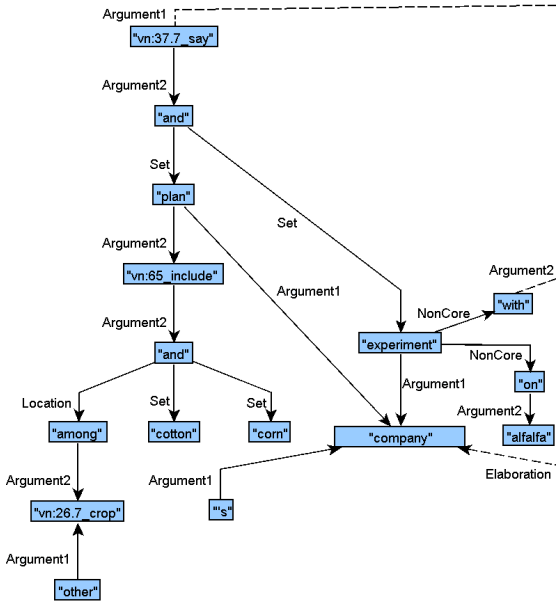


Figure 11: A sample subgraph extracted from a text graph

The evaluation shows that sentence packaging can be interpreted as a community detection problem since community detection algorithms aim to identify densely connected subgraphs – which can be expected from sentential structures. The evaluation suggests that the

subgraphs obtained by community detection can be further improved by a post-processing stage, i.e., by descent search or PCA.

5.7 Statistical graph transducers

In order to project a packaged semantic structure (SemS or DSyntS, henceforth DSyntS) onto its corresponding surface syntactic structures (SSyntS) in the course of generation (where both DSyntSs and their corresponding SSyntSs are stored in the 14-column CoNLL'09 format (Hajič et al., 2009)), it is possible to train statistical generators to use instead of the handcrafted grammars described in Section 4. The objective is to increase the coverage of the generator, which is done at the expense of the quality of the outputs.

The following types of actions need to be performed by a statistical generator (see Figure 12 for an illustration of steps 1-5; see (Ballesteros et al., 2015) for publication on intermediate results of this research):

1. Project each node in the DSyntS onto its SSynS-correspondence. This correspondence can be a single node, as, e.g., *job*->[*NN*] (where *NN* is a noun), or a subtree (hypernode, known as syntagma in linguistics), as, e.g., *time*->[*DT NN*] (where *DT* is a determiner and *NN* a noun, as in *the time*) or *create*>[*VAUX VAUXVB IN*] (where *VAUX* is an auxiliary, *VB* a full verb and *IN* a preposition, as in *to have been created*). In formal terms, we assume any SSyntS-correspondence to be a hypernode with a cardinality ≥1.
2. Generate the correct lemma for the nodes in SSyntS that do not have a 1:1 correspondence with an origin DSyntS node (as *DT* and *VAUX* above).
3. Establish the dependencies within the individual SSyntS-hypernodes.
4. Establish the dependencies between the SSyntS-hypernodes (more precisely, between the nodes of different SSyntS-hypernodes) to obtain a connected SSyntS-tree. For Spanish, after the DSyntS–SyntS we apply transition rules for the generation of relative pronouns that are implied by the SSyntS. Since we cannot count on the annotation of coreference in the training data, we do not treat other types of referring expressions. The lemmas of nodes with 1:1 correspondence are the same in both structures.
5. Establish the order between words given the surface-syntactic structure.
6. Inflect all the words.

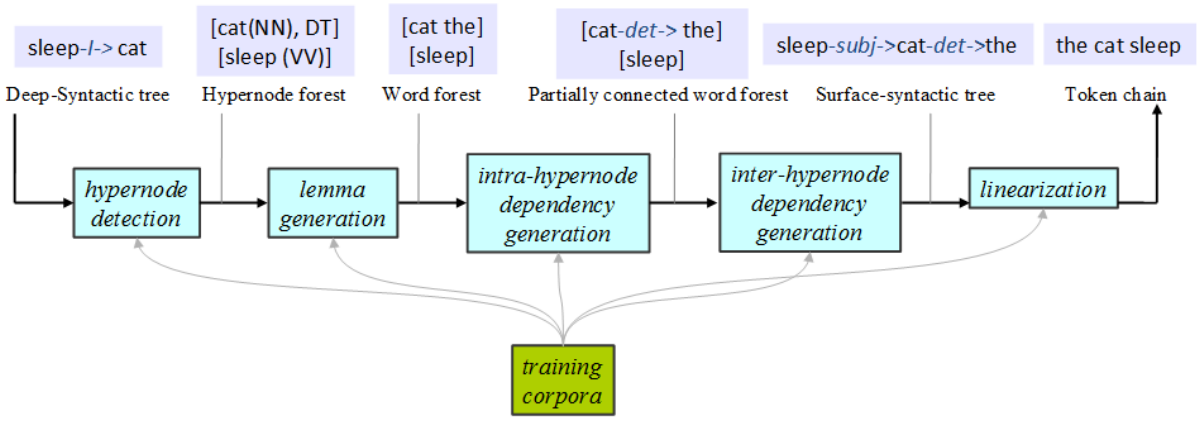


Figure 12: Workflow of the data-driven generator.

As it is the case for the rule-based generator, our planned statistical generators are based on the MTT (Meaning-to-Text) model.

Another strand of research in the context of this task concerns the development of a novel stochastic linearisation strategy. In contrast to current state-of-the-art linearisers, our implementation will take the distinctive features of different types of syntactic dependency relations into account. The development of the lineariser is still at a very early stage and will be discussed in the final deliverable.

For all statistical modules, training data is needed. Taking advantage of the growing availability of multilingual treebanks annotated with Universal Dependencies, the UD V2.0 treebank, as released in the context of the CoNLL 2017 shared task on multilingual dependency parsing (Zeman 2018), was used. UPF created a new multilingual dataset derived from the parsing data and organised an international Shared Task on Multilingual Surface Realization,[23] to the first edition of which 8 teams participated (Mille, et al. 2018).[24] A subset of ten languages was selected that contains the necessary part-of speech and morphological tags for the Shallow Track: Arabic, Czech, Dutch, English, Finnish, French, Italian, Portuguese, Russian and Spanish. Three of these languages, namely English, French and Spanish were used also for the Deep Track. In order to create this Deep dataset, the V4Design analysis pipeline was used; the parts of the dataset relevant to V4Design and their creation will thus be described in detail in D3.3 (M18, June 2019).

During the second part of the V4Design project, UPF will organise one or two more editions of the shared task and will train their own statistical generators on the newly created data for the V4Design languages.

[23] http://taln.upf.edu/pages/msr2018-ws/SRST.html

[24] The number of participants may seem low, but it has been one of the most successful tasks in Natural Language Generation in the past 10 years.

6 CONCLUSIONS

This document reports on the progress and achievements of tasks T5.4 (Ontology-based text planning) and T5.5 (Multilingual explanatory text generation for the 3D objects) during the first half of the project.

We first describe the approach and experiments with respect to summarisation carried out in the framework of the project, applicable for both extractive and abstractive summarisation, together with a first positive evaluation of the results. With respect to text generation, we detail the general adopted approach for going from ontological representations to well-formed texts. The rule-based generation pipeline consists of DnS-based translation rules and a sequence of rule-based graph transducers, for which multilingual lexical resources have also been developed. We report on an automatic evaluation of this pipeline in English, with an improvement of over 8% of accuracy compared to the beginning of the project. We describe future experiments about data-driven transducers, using new multilingual datasets that have been compiled within WP3. We also report on the advanced strategy for sentence packaging, which has already been implemented.

In the second half of the project, for the text generation component, we will improve the graph-transduction grammar quality (English) and coverage (Greek, Spanish, German), and train new statistical modules. As far as summarisation is concerned, we will adapt the algorithm to the specificities of the V4Design Knowledge Base, and will perform experiments on abstractive summarisation, by coupling the summarisation algorithms with the text generation module.

Extractive summarisation algorithms and the basic text generation pipeline have been integrated successfully in the V4Design architecture. There are no deviations with respect to the DoA to be reported on month 16.

7 REFERENCES

Alfonseca, Enrique, and Daniele Pighin. “HEADY: News headline abstraction through event pattern clustering.” *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* 2013. 1243-1253.

Arora, Sanjeev, Yingyu Liang, and Tengyu Ma. “Aa Simple but Tough-to-Beat Baseline for Sentence Embeddings.” *ICLR*, 2017.

Ballesteros, M., B. Bohnet, S. Mille, and L. Wanner. "Data-driven sentence generation with non-isomorphic trees." *Proceedings of NAACL-HLT.* ACL, 2015. 387-397.

Banarescu SDL, Laura, et al. "Abstract Meaning Representation for Sembanking." *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse.* 2013.

Banerjee, Siddhartha, Prasenjit Mitra, and Kazunari Sugiyama. “Multi-document abstractive summarization using ILP based multi-sentence compression.” *IJCAI International Joint Conference on Artificial Intelligence.* 2015. 1208-1214.

Barzilay, Regina, and Kathleen R. McKeown. “Sentence Fusion for Multidocument News Summarization.” *Computational Linguistics* 31, no. 3 (2005): 297-328.

Belz, A., M. White, D. Espinosa, E. Kow, D. Hogan, and A. Stent. “The first surface realisation shared task: Overview and evaluation results.” *Proceedings of the 13th European workshop on natural language generation.* Nancy, 2011.

Blondel, V., J.L. Guillaume, R. Lambiotte, and E. Lefebvre. "Fat unfolding of communities in large networks." *Journal of statistical mechanics: theory and experiment*, 2008.

Bohnet, B., and L. Wanner. “Open Source Graph Transducer Interpreter and Grammar Development Environment.” *Proceedings of LREC.* Valletta, 2010.

Bohnet, B., L. Wanner, S. Mille, and A. Burga. "Broad Coverage Multilingual Deep Sentence Generation with a Stochastic Multi-Level Realizer." *Proceedings of the 23rd International Conference on Computational Linguistics.* Beijing, 2010.

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov. “Enriching Word Vectors with Subword Information.” *Transactions of the Association for Computational Linguistics* 5 (2017): 135-146.

Bouayad-Agha, N., G. Casamayor, S. Mille, and L. Wanner. "Perspective-oriented generation of football match summaries: Old tasks, new challenges." *TSLP* 9, no. 2 (2012): 1-31.

Cheung, Jackie Chi Kit, and Gerald Penn. “Unsupervised Sentence Enhancement for Automatic Summarization.” *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* Doha, Qatar: Association for Computational Linguistics, 2014. 775-786.

Delli Bovi, Claudio, and Alessandro Raganato. “Sew-Embed at SemEval-2017 Task 2: Language-Independent Concept Representations from a Semantically Enriched Wikipedia.” *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017).* Association for Computational Linguistics (ACL), 2018. 261-266.

Elsner, Micha, and Deepak Santhanam. “Learning to fuse disparate sentences.” *Proceedings of the Workshop on Monolingual Text-To-Text Generation* (Association for Computational Linguistics), no. June (2011): 54-63.

Filippova, Katja, and Michael Strube. “Sentence fusion via dependency graph compression.” *Proceedings of the Conference on Empirical Methods in Natural Language Processing.* Association for Computational Linguistics, 2008. 177-185.

Filippova, Katja, and Yasemin Altun. “Overcoming the Lack of Parallel Data in Sentence Compression.” *Emnlp.* 2013. 1481-1491.

Flanigan, J., C. Dyer, N. Smith, and J. Carbonell. "Generation from abstract meaning representation using tree transducers." *Proceedings of NAACL:HLT.* 2016. 731-739.

Gardent, C., A. Shimorina, S. Narayan, and L. Perez-Beltrachini. "The WebNLG challenge: Generating text from RDF data." *Proceedings of the 10th International Conference on Natural Language Generation.* 2017. 124-133.

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

Gatt, A., and E. Krahmer. "Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation." *Journal of Artificial Intelligence Research, 61*, 2018: 65-170.

Gerani, Shima, Giuseppe Carenini, and Raymond T. Ng. “Modeling content and structure for abstractive review summarization.” *Computer Speech and Language*, 8 7 2015.

Gerani, Shima, Yashar Mehdad, Giuseppe Carenini, Raymond T Ng, and Bita Nejat. “Abstractive Summarization of Product Reviews Using Discourse Structure.” *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, (EMNLP 2014).* 2014. 1602-1613.

Hermann, Karl Moritz, et al. “Teaching Machines to Read and Comprehend.” *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1.* Cambridge, MA, USA: MIT Press, 2015. 1693-1701.

Johansson, R., and P. Nugues. "Extended constituent-to-dependency conversion for English." *Proceedings of the 16th Nordic Conference of Computational Linguistics (NODALIDA 2007).* 2007.

Joliffe, I. “Principal component analysis and factor analysis.” *Principal component analysis, Springer*, 1986: 115-128.

Karypis, G., and V. Kumar. "Multilevel k-way hypergraph partitioning." *VLSI design 11 (3)*, 2000: 285-300.

Khan, Atif, Naomie Salim, and Yogan Jaya Kumar. “A framework for multi-document abstractive summarization based on semantic role labelling.” *Applied Soft Computing Journal* (Elsevier) 30 (5 2015): 737-747.

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

Li, Peng, Tom Weidong Cai, and Heng Huang. “Weakly Supervised Natural Language Processing Framework for Abstractive Multi-Document Summarization.” Edited by James Bailey, et al. *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM).* ACM, 2015. 1401-1410.

Lin, C Y. “Rouge: A package for automatic evaluation of summaries.” *Proceedings of the workshop on text summarization branches out (WAS 2004)*, no. 1 (2004): 25-26.

Liu, Fei, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. “Toward Abstractive Summarization Using Semantic Representations.” *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015: 1077-1086.

Mann, William C, and Sandra A Thompson. “Rhetorical Structure Theory: A Frarnework for the Analysis of Texts.” *IPrA Papers in Pragmatics* 1 (1987).

Mehdad, Y, G Carenini, and R T Ng. “Abstractive summarization of spoken and written conversations based on phrasal queries.” *52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference* 1 (2014): 1220-1230.

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

Meyers, A., Reeves, R., C. Macleod, R. Szekely, V. Zelinska, B. Young, and R. Grishman. "The NomBank project: An interim report." *Proceedings of HLT-NAACL 2004 workshop: Frontiers in corpus annotation.* Boston, 2004.

Mille, S., A. Belz, B. Bohnet, Y. Graham, E. Pitler, and L. Wanner. "The First Multilingual Surface Realisation Shared Task (SR'18): Overview and Evaluation Results." *Proceedings of the 1st Workshop on Multilingual Surface Realisation.* Melbourne: ACL, 2018. 1-12.

Mille, S., M. Ballesteros, A. Burga, G. Casamayor, and L. Wanner. “Multilingual natural language generation within abstractive summarization.” *CEUR Workshop Proceedings.* 2016.

Mille, S., R. Carlini, A. Burga, and L. Wanner. "FORGe at SemEval-2017 Task 9: Deep sentence generation based on a sequence of graph transducers." *Proceedings of the 11th International Workshop on Semantic Evaluation.* Vancouver, 2017. 920-923.

Mille, S., R. Carlini, I. Latorre, and L. Wanner. "UPF at EPE 2017: Transduction-based Deep Analysis." *Shared Task on Extrinsic Parser Evaluation (EPE 2017).* 2017. 80-88.

Moro, Andrea, and Roberto Navigli. “SemEval-2015 Task 13: Multilingual All-Words Sense Disambiguation and Entity Linking.” 2015.

Nallapati, Ramesh, Bowen Zhou, C'icero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. “Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond.” Edited by Yoav Goldberg and Stefan Riezler. *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016.* ACL, 2016. 280-290.

Navigli, R., and S.P. Ponzetto. “BabelNet: Building a very large multilingual semantic network.” *Proceedings of the 48th annual meeting of the association for computational linguistics.* Uppsala, 2010. 216-225.

Nayak, N., D. Hakkani-Tur, M. Walker, and L. Heck. "To plan or not to plan? discourse planning in slot-value informed sequence to sequence models for language generation." *Proceedings of Interspeech.* 2017.

Otterbacher, Jahna, Gunes Erkan, and Dragomir R. Radev. “Biased LexRank: Passage retrieval using random walks with question-based priors.” *Information Processing and Management* 45, no. 1 (2009): 42-54.

Paulus, Romain, Caiming Xiong, and Richard Socher. “A Deep Reinforced Model for Abstractive Summarization.” *6th International Conference on Learning Representations (ICLR)*, 5 2018.

Perez-Beltrachini, L., and C. Gardent. "Learning embeddings to lexicalise rdf properties." *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics.* 2016. 219-228.

Perone, Christian S., Roberto Silveira, and Thomas S. Paula. “Evaluation of sentence embeddings in downstream and linguistic probing tasks.” 2018.

Pighin, Daniele, Enrique Alfonseca, Marco Cornolti, and Katja Filippova. “Modelling Events through Memory-based , Open-IE Patterns for Abstractive Summarization.” *Acl*, 2014: 892-901.

Rambow, O., and T. Korelsky. “Applied text generation.” *Proceedings of the third conference on Applied natural language processing.* Association for Computational Linguistics, 1992. 40-47.

Rodriguez-Garcia, M.A. , and R. Hoehndorf. "Inferring ontology graph structures using OWL reasoning." *BMC Bioinformatics 19(7)*, 2018.

Rush, Alexander M, Sumit Chopra, and Jason Weston. “A Neural Attention Model for Abstractive Sentence Summarization.” *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015.* The Association for Computational Linguistics, 2015. 379--389.

Sandhaus, Evan. “The new york times annotated corpus.” *Linguistic Data Consortium, Philadelphia.* 2008.

Schuler, K.K. *VerbNet: A broad-coverage, comprehensive verb lexicon.* Univeristy of Pennsylvania, 2005.

See, Abigail, Peter J Liu, and Christopher D Manning. “Get To The Point: Summarization with Pointer-Generator Networks.” Edited by Regina Barzilay and Min-Yen Kan. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, {ACL} 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers.* Association for Computational Linguistics, 2017. 1073-1083.

Takase, Sho, Jun Suzuki, Naoaki Okazaki, Tsutomu Hirao, and Masaaki Nagata. “Neural Headline Generation on Abstract Meaning Representation.” Edited by Jian Su, Xavier

Carreras and Kevin Duh. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, {EMNLP} 2016, Austin, Texas, USA, November 1-4, 2016.* The Association for Computational Linguistics, 2016. 1054-1059.

Thadani, Kapil, and Kathleen Mckeown. “Supervised Sentence Fusion with Single-Stage Inference.” *Proc. IJCNLP 2013*, no. October (2013): 1410-1418.

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

Wang, Yafang, Zhaochun Ren, Martin Theobald, Maximilian Dylla, and Gerard de Melo. “Summary generation for temporal extractions.” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).* Springer, Cham, 2016. 370-386.

Wanner, L., and et al. "Ontology-centered environmental information delivery for personalized decision support." *Expert Systems with Applications* 42, no. 12 (2015): 5032-5046.

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

Zeman, D. et al. “CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies.” *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies.* 2018. 1-21.

8 APPENDIX A

Guidelines followed for the annotation of summaries with meanings in BabelNet:

1. Only annotate text spans consisting of a single or multiple consecutive words, e.g. do not annotate "Blue iced juice" as "blue juice", even if there is a synset for it.
2. If a suitable meaning is found for a multiword, do not annotate any meanings for parts of the multiword, e.g. if a synset is found for "emergency break", do not annotate "emergency" nor "break".
3. Do not annotate parts of a multiword NE even if no synset is found for it, e.g. do not annotate "Ponte" nor "Angeli" even if there is no synset for "Ponte Angeli".
4. Given a text span s and a set of synsets M returned by BabelNet after looking up s:
    - As a rule of thumb, annotate s with any suitable meanings in M.
    - Multiple synsets are allowed if they are all judged to be correct.
    - Words in s can be replaced by lemmas if necessary, e.g. look up "euro" instead of "euros".
    - If a NE synset is not in M but can be easily and unequivocally inferred from the context, annotate s with it, e.g "cruise liner" may be inferred to refer to the specific ship "Costa Concordia" in addition to the generic concept "cruise liner".
5. Do not attempt to annotate complex metaphors or inferred concepts not in M, e.g. do not try to guess if "the best feeling" refers to a specific emotion such as love, do not annotate "losing side" with "loser".
6. Words with multiple POS interpretations -> annotate correct meanings belonging to any of possible POS, e.g. annotate verbal and adjectival synsets for words like "found", "convicted", annotate adjectival and adverbial synsets for "fast".
7. Annotate meanings for quantities, units, currencies, time periods, and percentages, e.g. in "10% of 100€" annotate "10", "%", "100" and "€".
8. Annotate the following types of words only:
    - Nouns: Yes.
    - Main verbs: Yes.
    - Adverbs: Yes.
    - Adjectives: Yes.
    - Determiners and pronouns:
        - Quantifier: Yes (few, fewer, little, many, much, more, most, some, any).
        - Number: Yes.
        - Article: No (a/an, the).
        - Demonstrative: No (this, that, these, those).
        - Possessive: No (my, your, his, her, its, our, their, x's).
        - Relative and interrogative: No (who, whom, whose, which, what, that).
        - Personal: No (I, me, she, they).
        - Reflexive and reciprocal: No (himself, each other).
        - Indefinite: No (one, other, another, no one, anybody, nothing, everything, something, someone, whatever, whoever, none, all, both, either, such, each, etc.).

- Auxiliary verbs:
  - Copula: No.
  - Modal: No.
  - Tense: No .
  - Aspect: No.
- Idioms, temporal expressions: No (Just as, In the light of).
- Conjunctions: No.
- Non-governed prepositions: No (in 2005).

9. Annotate using the format synset_id0-"annotated words" or synset_id0|synset_id1|...-"annotated words" if multiple correct meanings are found.