Underspecified Universal Dependency Structures as Inputs for Multilingual Surface Realisation

Simon Mille
Universitat Pompeu Fabra
Barcelona, Spain
simon.mille@upf.edu

Anja Belz
University of Brighton
Brighton, UK
a.s.belz@brighton.ac.uk

Bernd Bohnet
Google Inc.
London, UK
bohnetbd@google.com

Leo Wanner
ICREA and Universitat Pompeu Fabra
Barcelona, Spain
leo.wanner@upf.edu

Abstract

In this paper, we present the datasets used in the Shallow and Deep Tracks of the First Multilingual Surface Realisation Shared Task (SR'18). For the Shallow Track, data in ten languages has been released: Arabic, Czech, Dutch, English, Finnish, French, Italian, Portuguese, Russian and Spanish. For the Deep Track, data in three languages is made available: English, French and Spanish. We describe in detail how the datasets were derived from the Universal Dependencies V2.0, and report on an evaluation of the Deep Track input quality. In addition, we examine the motivation for, and likely usefulness of, deriving NLG inputs from annotations in resources originally developed for Natural Language Understanding (NLU), and assess whether the resulting inputs supply enough information of the right kind for the final stage in the NLG process.

1 Introduction

There has long been an assumption in Natural Language Generation (NLG) that surface realisation can be treated as an independent subtask for which stand-alone, plug-and-play tools can, and should, be created. Early surface realisers such as KPML (Bateman, 1997) and FUF/Surge (Elhadad and Robin, 1996) were ambitious, independent surface realisation tools for English with wide grammatical coverage. However, the question of how the NLG components addressing the stage before surface realisation were supposed to put together inputs of the level of grammatical sophistication required by such tools was never quite resolved. The success of SimpleNLG (Gatt and Reiter, 2009) which had much reduced grammatical coverage, but accepted radically simpler inputs demonstrated the importance of this issue.

The recently completed first Multilingual Surface Realisation Task (SR'18) (Mille et al., 2018) used for the first time inputs derived from the Universal Dependencies (UDs) (de Marneffe et al., 2014), a framework which was devised with the aim of facilitating cross-linguistically consistent grammatical annotation, and which has grown into a large-scale community effort involving more than 200 contributors, who have created over 100 treebanks in over 70 languages between them.[1] UDs provide a more general and potentially flexible input representation for surface realisation (SR). However, their use for NLG has not so far been demonstrated.

In this paper, we present the UD datasets used in the Shallow and Deep Tracks in SR'18, describe the precise conversion processes that were applied to them, and provide an assessment of their quality. Furthermore, we examine (a) the SR task in general, (b) the motivation for, and likely usefulness of, the derivation of NLG inputs from annotations in resources developed for Natural Language Understanding (NLU), (c) whether the resulting inputs supply enough information of the right kind for the final stage in the NLG process, and more tentatively, (d) what role SR is likely to play in the future in the NLG context.

Section 2 presents related work; Section 3 describes the datasets used in the two SR'18 tracks, and Section 4 provides a more procedural account of how the datasets were generated. Section 5 assesses the quality of the obtained representations,

[1] http://universaldependencies.org/

while Section 6 discusses their suitability for SR and NLG more generally. Some conclusions are presented in Section 7.

2 Background

With the advent of large-scale treebanks and statistical NLG, surface realisation research turned to the use of treebank annotations, processed in various ways, as inputs to surface realisation. Annotation/sentence pairs constitute the training data, and similarity to the original sentences in the treebank is the main measure of success. A lot of this work used inputs derived from the Wall Street Journal corpus with varying amounts of information removed from the parse-tree annotations (Langkilde-Geary, 2002; Nakanishi et al., 2005; Zhong and Stent, 2005; Cahill and van Genabith, 2006; White and Rajkumar, 2009). Because of the variation among inputs, results were not entirely comparable. The first Surface Realisation Shared Task (SR'11) (Belz et al., 2011) was thus conceived with the aim of developing a common-ground input representation that would make different systems, for the first time, directly comparable. SR'11 used shallow and deep inputs for its two respective tracks, both derived from CoNLL'08 shared task data, which was in turn derived from the WSJ Corpus by automatically converting the corresponding Penn TreeBank parse trees to dependency structures (Surdeanu et al., 2008). While dependency structures offer a more flexible input structure and statistical systems, in principle, offer more robustness, the uptake of such systems as components in embedding NLG systems has been very limited.

Meanwhile, many of the more applied strands of NLG research have tended to bypass an explicit interface to surface realisation, instead mapping directly from more abstract representations of meaning to surface text. Recently, mostly under the aegis of the Generation Challenges series of shared tasks, several large-scale datasets have been made available that pair surface text with more abstract structured inputs, including:

- Weather forecast generation (Weather) dataset (Liang et al., 2009): time series from weather-related measurements;
- Abstract Meaning Representation (AMR) dataset (May and Priyadarshi, 2017): abstract predicate-argument graphs that cover several genres;
- WebNLG dataset (Gardent et al., 2017): DBpedia triples covering properties of 15 DBpedia categories;
- E2E dataset (Novikova et al., 2017): attribute-value pairs covering 8 properties related to the restaurant domain.

In all these datasets, the input structures are aligned with English sentences that match their contents. In the case of inputs coming from structured data (e.g. WebNLG, E2E, Weather, above), multiple sentences are generally paired with each input, whereas for the inputs coming from data initially annotated for Natural Language Understanding tasks (SR'11, AMR), only one reference per input is available. Both types of shared tasks (reusable surface realisation vs. task-specific generation) have been successful in terms of participation levels, and both have sizeable research communities behind them, but trainable surface realisation as a stand-alone subtask still has a way to go in terms of demonstrating its practical applicability.

3 The SR'18 Data

The First Multilingual Surface Realisation Shared Task (SR'18) ran from December 2017 to July 2018. As in SR'11, the shared task comprised two tracks with different levels of difficulty: a Shallow Track, starting from syntactic structures in which word order information has been removed and tokens have been lemmatised, and a Deep Track, which starts from more abstract structures from which, additionally, functional words (in particular, auxiliaries, functional prepositions and conjunctions) and surface-oriented morphological information have been removed.

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 et al., 2017), was used. 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. Starting from UD structures as they appear in the treebanks, Shallow and Deep inputs

1	The	the	DET	DT	Definite=Def|PronType=Art	2	det
2	third	third	ADJ	JJ	Degree=Pos|NumType=Ord	5	nsubj_pass
3	was	be	AUX	VBD	Mood=Ind|Number=Sing|Person=3|Tense=Past|VerbForm=Fin	5	aux
4	being	be	AUX	VBG	VerbForm=Ger	5	aux_pass
5	run	run	VERB	VBN	Tense=Past|VerbForm=Part|Voice=Pass	0	root
6	by	by	ADP	IN	_	8	case
7	the	the	DET	DT	Definite=Def|PronType=Art	8	det
8	head	head	NOUN	NN	Number=Sing	5	obl
9	of	of	ADP	IN	_	12	case
10	an	a	DET	DT	Definite=Ind|PronType=Art	12	det
11	investment	investment	NOUN	NN	Number=Sing	12	compound
12	firm	firm	NOUN	NN	Number=Sing	8	nmod
13	.	.	PUNCT	.	_	5	punct

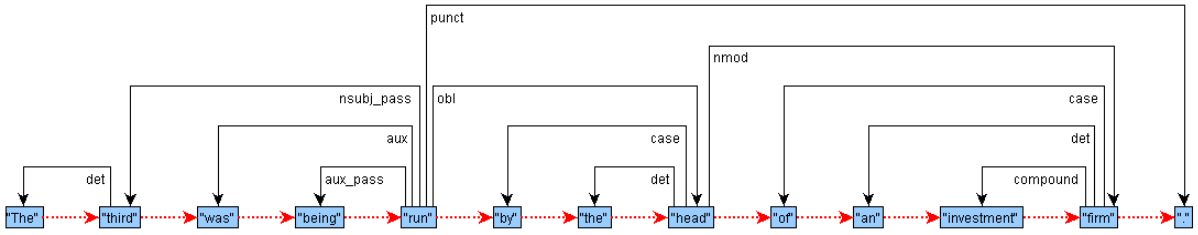


Figure 1: A sample UD structure in English (top: CoNLL-U, bottom: graphical)

1	the	_	DET	DT	Definite=Def|PronType=Art	2	det
2	third	_	ADJ	JJ	Degree=Pos|NumType=Ord	3	nsubj_pass
3	run	_	VERB	VBN	Tense=Past|VerbForm=Part|Voice=Pass	0	root
4	be	_	AUX	VBD	Mood=Ind|Number=Sing|Person=3|Tense=Past|VerbForm=Fin	3	aux
5	be	_	AUX	VBG	VerbForm=Ger	3	aux_pass
6	head	_	NOUN	NN	Number=Sing	3	obl
7	.	_	PUNCT	.	_	3	punct
8	by	_	ADP	IN	_	6	case
9	the	_	DET	DT	Definite=Def|PronType=Art	6	det
10	firm	_	NOUN	NN	Number=Sing	6	nmod
11	an	_	DET	DT	Definite=Ind|PronType=Art	10	det
12	investment	_	NOUN	NN	Number=Sing	10	compound
13	of	_	ADP	IN	_	10	case

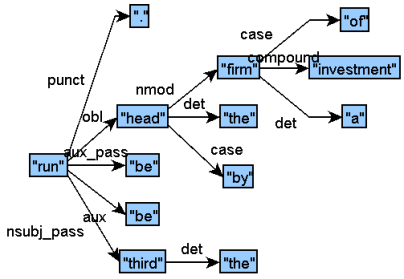


Figure 2: Shallow input (Track 1) derived from UD structure in Figure 1. (top: CoNLL-U, bottom: graphical)

were created automatically for the two tracks. The inputs for each language and track were released in the CoNLL-U format[2], together with parallel files that contain a reference sentence for each input. Figures 1, 2 and 3 show a sample original UD annotation for English, and the corresponding inputs for the Shallow and Deep Tracks, respectively, in the 10-column CoNLL-U format (the last two columns generally do not contain information and are thus omitted) and in graphical format. The training, development and test data are available at http://taln.upf.edu/pages/msr2018-ws/SRST.html#data.

[2] http://universaldependencies.org/format.html

3.1 Shallow inputs

The Shallow Track input structures are unordered syntactic trees with all the words of the sentence replaced with their lemmas, and labelled with their part-of-speech tags and the morphological information associated with each node. These structures are thus genuine UD structures, with only two differences: first, in the original CoNLL-U format, consecutive lines contain words that are also consecutive in the sentence, whereas in the SR'18 Shallow structures, no order information is available; second, original UD structures contain both lemmas and inflected forms, while only the former are available in the SR'18 structures. Fig-

1	third	_	ADJ	_	Degree=Pos	2	A2
2	run	_	VERB	_	Tense=Past|Aspect=Progr	0	ROOT
3	head	_	NOUN	_	Number=Sing|Definiteness=Def	2	A1
4	firm	_	NOUN	_	Number=Sing|Definiteness=Indef	3	A2
5	investment	_	NOUN	_	Number=Sing	4	AM

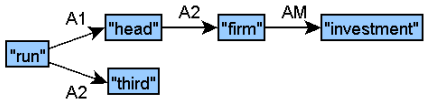


Figure 3: Deep input (Track 2) derived from UD structure in Figure 1. (left: CoNLL-U, righ: graphical)

ures 1 and 2 show an original UD structure and a SR'18 Shallow input, respectively.

3.2 Deep inputs

The Deep Track input structures are trees that contain only content words linked by predicate-argument edges, in the PropBank/NomBank (Palmer et al., 2005; Meyers et al., 2004) fashion.

The Deep inputs can be seen as closer to a realistic application context for NLG systems, in which the component that generates the inputs presumably would not have access to syntactic or language-specific information. At the same time, we used only information found in the UD structures to create the Deep inputs, and tried to keep their structure simple. In Deep inputs, words are not disambiguated, full (semantically loaded) prepositions may be missing, and some argument relations may be underspecified or missing. The next two subsections provide more details about the Deep nodes and edge labels.

3.2.1 Deep nodes

In contrast to the Shallow structures, which contain all the (lemmatised) words of the original sentence, the Deep structures do not contain functional words that can be inferred from another lexical unit or from the syntactic structure (such as bound prepositions and conjunctions), or can be represented as a feature on another node (such as auxiliaries, modals, or determiners). For instance (see also Figure 1 for the first four examples):

- the preposition *by* in the sentence *The third was being run by the head of an investment firm* is bound to the English agentive dependency in a passive construction;
- the preposition *of* in *the head of an investment firm* is bound to the noun *head*, and indicates the presence of its second argument;
- *being* in *was being run* is a passive voice marker, while *was* is part of the marker for the progressive aspect, associated with the verb *run*;
- *the* in *the head* can be seen as a marker for nominal definiteness;
- the conjunction (complementiser) *that* in, e.g., *I demand that you apologise*, appears because it connects a finite verb *apologise* as an argument of another verb *demand*.

Meaningful functional words such as auxiliaries and determiners are represented as attribute/value pairs associated with the relevant nodes, as, e.g., the *Aspect* feature on *run* in Figure 3.[3] Features are also used to encode information such as verbal tense or nominal number, which are needed for realisation. On the other hand, implicit nodes that have a role in the sentence are explicit in the Deep input: for instance, dummy pronoun nodes for the subject if an originally finite verb has no first argument and no available argument to build a passive or, for a pro-drop language such as Spanish, dummy pronouns when the first argument of a verb is missing.

3.2.2 Deep edge labels

In Deep inputs, content words are linked by predicate-argument labels in the PropBank/NomBank (Palmer et al., 2005; Meyers et al., 2004) fashion, that is, there are core (A1, A2, etc.) and non-core (AM) labels. Additional labels such as LIST or NAME have been added in order to connect all the elements within each sentence; see Table 1 for the inventory of relations. This subsection details the main features of the dependencies at this level.

First of all, the first argument is always labeled as A1, that is, there is no external argument A0, as can be found in PropBank and NomBank. For instance, both the verbs *fall* and *fancy* have a relation A1 with their first argument (*the ball*$_{A1}$ *falls*, *Martin*$_{A1}$ *fancies Chloe*), even though according to PropBank, the latter has an external argument (*Martin*$_{A0}$ *fancies Chloe*).

[3] For the available universal features set, see http://universaldependencies.org/u/feat/index.html.

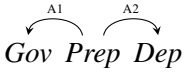
Deep label	Description	Example
A1, A2, ..., A6	nth argument of a predicate	fall→ the ball
A1INV, ..., A6INV	nth inverted argument of a predicate	the ball→ fall
AM/AMINV	(i) none of governor or dependent are argument of the other (ii) unknown argument slot	fall→ last night
LIST	List of elements	fall→ [and] bounce
NAME	Part of a name	Tower→ Eiffel
DEP	Undefined dependent	N/A

Table 1: Deep labels

Second, in order to maintain the tree structure and account for some cases of shared arguments, there can be inverted argument relations. Consider, for instance, the difference between *the ball*$_{A1}$ *falls*, in which *ball* is the first argument of *fall*, and *the falling*$_{A1INV}$ *ball*, in which *fall* has the 'inverted first argument' label, which means that *ball* is actually the first argument of *fall*. Inverted relations are used also in relative clauses, as e.g., in *the ball that falls*$_{A1INV}$; in such a case, the subject (or object) relative pronoun does not appear in the structure, and the antecedent can thus be shared by two predicates, as An and AnINV.

Third, all modifier edges are assigned the same generic label AM. When an edge is underspecified, that is, when there is no predicate-argument relation between two connected nodes, or when there is a predicate-argument relation between two nodes, but it is not known which one specifically, the AM or AMINV label is used. For instance, in the case of nominal adverbials such as *last night*, the relation between the governing predicate and *night* is set as an AM. One of the most productive uses of the AM label is for prepositional groups or subordinate clauses, in which UD establishes a direct dependency between the content words that actually does not exist: *he will leave*$_{Gov}$ *after he finishes*$_{Dep}$ *his work*. In this case, *finish* is established as an AM of *leave*; note that this does not prevent the actual argumental structure from being recovered. Indeed, an edge AM that links a Governor with a Dependent and its A2INV preposition:

is equivalent to the preposition being a predicate and the Governor and Dependent being its first and second argument respectively:



Fourth, there are some edge labels that are not related to predicate-argument relations: coordinated elements are linked by a dedicated edge LIST, compound named entities are linked by the edge NAME, and the edge label DEP is used in case of an unknown relation between two elements. Finally, each argument relation is unique for each predicate: if a predicate has an A2 dependent, it cannot have another A2 dependent, and it cannot be the A2INV dependent of another predicate.

In Section 4.2 below, we describe how the mapping between surface dependencies and predicate-argument labels was performed.

4 Generating the datasets

Following on from the more declarative description of Shallow and Deep inputs in the previous section, this section describes how those input were automatically created from the CoNLL'17 data.

4.1 Adaptation of the original UD structures

For the input to the Shallow Track, the UD structures were processed as follows:

1. the information on word order was removed by randomised scrambling;
2. the words were replaced by their lemmas.

For the Deep Track, additionally:

3. Functional prepositions and conjunctions in argument position, i.e. prepositions and conjunctions that can be inferred from other lexical units or from the syntactic structure, were removed; prepositions and conjunctions retained in the Deep representation can be found under an A2INV dependency;
4. Definite and indefinite determiners, auxiliaries and modals were converted into attribute/value pairs, as are definiteness features, and the universal aspect and mood features;

	ar	cs	en	es	fi	fr	it	nl	pt	ru
train	6,016	66,485	12,375	14,289	12,030	14,529	12,796	12,318	8,325	48,119
dev	897	9,016	1,978	1,651	1,336	1,473	562	720	559	6,441
test	676	9,876	2,061	1,719	1,525	416	480	685	476	6,366

Table 2: SR'18 dataset sizes for training, development and test sets.

5. copulas were removed and their two arguments connected with each other;
6. Subject and object relative pronouns directly linked to the main relative verb were removed (and instead, the verb is linked to the antecedent of the pronoun), and dummy pronouns were added.
7. Edge labels were generalised using the labels presented in Section 3.2.2;
8. Surface-level morphologically relevant information as prescribed by syntactic structure or agreement (such as verbal finiteness or verbal number) was removed, whereas semantic-level information such as nominal number and verbal tense was retained;
9. Fine-grained POS labels found in some treebanks (see e.g. column 5 in Figure 2) were removed, and only coarse-grained ones were retained (column 4 in Figures 2 and 3).

4.2 Generation of the inputs

Shallow Track inputs were generated with a Python script from the original UD structures, which were simply scrambled and their words replaced with lemmas. During the conversion, sentences that contained dependencies that only make sense in an analysis context were filtered out (e.g. *reparandum*, or *orphan*); this amounted to around 1.5% of sentences for all languages on average. Table 2 summarises the final size of the datasets.

Deep Track inputs were then generated by automatically processing the Shallow Track structures using a series of graph-transduction grammars (Bohnet and Wanner, 2010) that cover steps 3–9 above (in a similar fashion to Mille et al. (2017)), while ensuring a node-to-node correspondence between the Deep and Shallow structures.

The graph-transduction grammars are rules that apply to a subgraph of the input structure and produce a part of the output structure. During the application of the rules, both the input structure (covered by the left-hand side of the rule) and the current state of the output structure at the moment of application of a rule (i.e., the right-hand side of the rule) are available as context. Figure 4 shows the rule that assigns deep dependency labels as found in the UD_lexicon dictionary; relative clause edges and adjectival modifiers are handled by different rules. The output structure in one transduction is built incrementally: the rules are all evaluated, the ones that have no right-hand side context and that match a part of the input graph are applied, and a first piece of the output graph is built; then the rules are evaluated again, this time with the right-hand side context as well, and another part of the output graph is built; and so on. The transduction is complete when no rule is left that matches the combination of the left-hand side and the right-hand side. At each iteration, the rules are first selected and then applied as a cluster, that is, the order in which they apply is not important. The only way to force a rule R2 to be applied after a rule R1 (for instance, for building edges after building the nodes) is to establish as R2's right-side condition elements built by R1: for instance, the rule shown in Figure 4 will apply only after the rules that build the nodes have applied, since the *?Xr* and *?Yr* are marked as right-side context (*rc:*). This allows the rules to be more generic and to combine with one another in an efficient way.

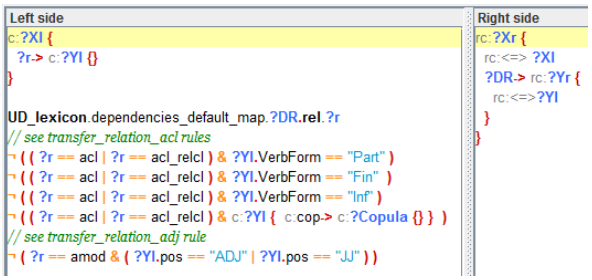


Figure 4: Sample graph transduction rule

Table 3 provides a summary of the graph-transduction grammars and rules for the mapping between surface-syntactic structures and UD-based semantic structures. The mapping is composed of three submodules. The pre-processing grammars are used to identify all nodes to be removed: it is easier and safer in the graph-transduction grammars to mark the nodes to be re-

Grammars	# rules	Description
Pre-processing	76	Identify nodes to be removed Identify verbal finiteness and tense
SSynt-Sem	120	Remove idiosyncratic nodes Establish correspondences with surface nodes Map UD labels to predicate-argument dependency labels (when possible) Predict predicate-argument dependency labels (when no direct mapping is available) Replace determiners, modality and aspect markers by attribute-value feature structures
Post-processing	60	Replace duplicated argument relations by best educated guess Identify remaining duplicated core dependency labels (for posterior debugging)

Table 3: Graph-transduction rules for producing the Deep inputs (counts include rules that simply copy node features, constituting about 40 per grammar).

moved (using solely positive conditions) and then to not generate them in the next step, than using rules that generate in one shot only the desired nodes, which would imply complex negative conditions. After this pre-processing, with the nodes that are to be removed marked, the core grammar (SSynt-Sem) takes care of establishing edge labels between the remaining nodes and associating the latter with attribute/value pairs. Most UD labels are mapped one-to-one to predicate-argument labels. In some cases only, the rules check the syntactic context of an edge in order to get a more precise label (e.g., a relative clause in which according to the structure and the UD label, we know that an argumental relation is holding: if the governor already has a first and a third arguments, the argumental relation is likely to be an A2). A post-processing grammar takes educated guesses in order to correct obvious labeling errors such as the duplication of an argument for a predicate. The unified cross-language annotation scheme of UD allows the large majority of the rules to be language-independent. Even though the annotations are not always consistent, adapting the grammars to a new language is relatively easy: most of the language-specific rules concern the processing of auxiliaries and modals, which have to be identified and mapped to the *Aspect* and *Mood* features.

5 Evaluation of the generated datasets

Since the processing applied to the Shallow inputs consists only in removing information and is very straightforward. It does not call for an evaluation. For the Deep Track, however, the changes are much more complex and the quality of the conversion needs to be assessed.

We evaluated the quality of the Deep inputs as follows. One of the authors manually annotated about 900 deep tokens (≈75 sentences) in each language (English, French and Spanish), by post-editing the automatically converted structures correcting any mistakes. Since the same person post-edited all three datasets, the resulting gold-standard is consistent across the languages, even though it does not allow for calculating inter-annotator agreement. Note that the annotation remains quite open with respect to some phenomena, for which several annotations are considered correct. For instance, AM relations are left underspecified when it is not clear what argument slot is concerned (e.g., appositions, parentheticals, verbal/nominal adverbials, etc.); some argumental relations are ambiguous and left as such: $N \rightarrow ADJ$ is sometimes A1INV, and the adjective is sometimes an argument of the noun; numbers (e.g., *ten thousand people*) and hours are left as they are in the original annotations.

Once post-edited, the reference structures are compared to the ones produced by the automatic mapping from UD structures, using the LAS evaluation method of Ballesteros et al. (2015), specifically designed to handle the comparison between non-isomorphic trees. Since part of the mapping consists of adding and/or removing nodes, it often happens that the gold-standard and predicted structures end up with a different number of nodes, which makes evaluation scripts based on a strict node-to-node comparison unusable. Table 4 shows the results of the evaluation. Quality is not the same across languages: while English structures obtain an LAS of 79.83, French is more than 6 points lower, and Spanish more than 12. Since, as mentioned in the previous subsection, the mapping grammars are largely language-independent, and since roughly the same efforts have been dedicated to each language, it is likely that the LAS numbers reflect the quality of the original UD annotation.

Note that during the evaluation, POS and lemma-

	LAS
English	79.83
French	73.43
Spanish	67.28

Table 4: Evaluation of the quality of the output structures (Labeled Attachment Scores - LAS).

tisation[4] errors are not corrected, but structural errors due to original tagging/lemmatising errors are counted. In other words, what is being evaluated is how correct the outputs are in terms of dependencies and labeling, rather than how well the transduction grammars perform. An error analysis showed that most dependency errors come from the AM relation, which is usually A1, A2, A1INV or A2INV in the reference structures. The systematic replacement of AM by one of these four labels always results in a drop of the LAS score. That is, in order to improve the quality of the structures, an improvement of the UD structures or a more fine-grained processing (which would imply a large number of rules and the use of detailed lexicons) would be needed.

The mapping grammars were released together with the SR'18 datasets;[5] they can process about 39 sentences per second on an average laptop. The resulting mapping tool allows for automatically annotating large amounts of data. The tool has recently been used to convert about 600,000 English sentences that had been automatically parsed with an off-the-shelf UD parser. This tool is also currently being tested as part of conceptual relation extraction pipelines in the framework of several EU projects (see Acknowledgements).

6 Discussion

The UD-derived input structures described in this paper were successfully used in the SR'18 Shared Task, which attracted the participation of eight teams. In the Deep Track, an attempt was made to remove from inputs, as far as possible, the kind of information that cannot come from a deeper level of abstraction, such as, e.g., an ontological representation. For instance, where it was not possible, or too risky, to predict an argument slot, it was left undefined (AM label). If, because the annotation did not allow for the distinction between the two, there was a choice between leaving too many syntactic elements or removing meaningful words, the latter option was chosen. In this way, the Deep representations are much closer to the kind that might be used in a generation pipeline that starts from structured data,[6] and the tools trained on the present data can potentially be used in an applied NLG pipeline. On the other hand, the inputs can be considered less informative than those used in SR'11, in which only *that*-complementisers and *to*-infinitives were removed, and predicate-argument labels for nouns and verbs were fully specified (since they came from the NomBank and PropBank manually validated annotations). However, as is the case with the tectogrammatical layer of the Prague Dependency Treebank (Böhmová et al., 2005), in PropBank and NomBank no distinction is made between full and functional ('semantic') prepositions.[7] In contrast to AMRs, the Deep inputs do contain tense, number and definiteness information, but links to named entity databases or OntoNotes labels are not provided; in other words, the nodes are not disambiguated. The other difference to AMRs in terms of specificity is that the annotation of shared arguments is incomplete in SR'18 (an argument can only be shared by two predicates in SR'18 Deep inputs), and that the non-core relations are not typed. In terms of abstraction level, AMRs abstract the labels of nominal vs. verbal events, which is not done in the SR'18 dataset. In the SR'18 Shallow inputs, the removal of word order information is problematic for named entities, n-ary coordinations and punctuations, because it is not always possible to reconstruct the word order based on the dependencies. In order to cope with this, in SR'11, the components of named entities were numbered according to their original order, specific features encoded the bracketing information for punctuation signs, and coordinations were hierarchical, with each conjunct being a dependent of the previous conjunct.

As a result, the Deep input representation is a compromise between correctness and adequacy in a generation setup. Indeed, the conversion of the

[4] As in lexical reflexive verbs in French and Spanish; e.g. *aburrirse* 'to be bored' in Spanish, can end up with the lemma *aburrir*, that is, without the reflexive marker.

[5] http://taln.upf.edu/pages/msr2018-ws/SRST.html#data

[6] Note, however, that creating the Deep input structures from structured data would be far from trivial, since it would imply mapping given properties onto words as used in the UD datasets.

[7] This subcategorisation information can be partially derived from PropBank and NomBank, as done for the Deep Syntactic structures of Meaning-Text Theory (Ballesteros et al., 2015; Mille and Wanner, 2015).

UD structures into predicate-argument structures depends not only on the mapping process, but also on the availability of the information in the original annotation, even though it falls short in some cases: UD structures were not conceived for NLG applications, which is why using them in an NLG context presents considerable challenges. The underspecified UD structures created for the SR'18 Shared Task are perhaps as close as we can get to NLG-like meaning representations if all we have to construct inputs are parsing annotations from treebanks. To get even closer, the inputs would have to be enriched from additional sources of information, such as subcategorisation information, as found in PropBank or Ontonotes.

We may even be moving away from a situation where we have to rely on treebanks to obtain NLG inputs at reasonable cost, as the success of using fully automatically parsed data mentioned above shows. It is conceivable that a future shared task in NLG will involve paired (structured) data and text, plus an automatically created intermediate level of representation comprising underspecified UD (UUD) structures enriched with additional information obtained from the structured data level. This would correspond to three linked tracks (data-to-text, data-to-UUD, and UUD-to-text) where one track is the end-to-end task, and the other two tracks are subtasks that can be combined to solve the end-to-end task, similar to the GREC'10 shared task competition (Belz and Kow, 2010).

Or it could be argued, perhaps controversially still, that the days of structured linguistic representations in NLG are numbered anyway. The rapid development and spread of highly successful neural approaches to diverse NLG tasks, and the limited success so far of attempts to inject linguistic knowledge directly into neural networks, certainly lends some strength to this point of view. In the meantime, the above tripartite shared-task structure has the potential to accommodate both systems that map directly from data to text without structured representations, and two-component systems with a surface realiser as the second component.

7 Conclusion

In this paper, we have provided a detailed description of how the inputs to the Shallow and Deep Tracks at the First Multilingual Surface Realisation Task were created by automatically converting annotated sentences from Universal Dependency treebanks V2.0 into Shallow and Deep Track inputs. The important contribution here is the process for creating Deep inputs, where we approximate the kind of abstract meaning representations used in native NLG tasks. This is not a simple matter of applying a few replacement rules (as it is for the Shallow inputs) with predictably correct results. To assess the quality of the Deep inputs, we conducted an evaluation that showed a labelled agreement score (LAS) of about 80 with human-corrected equivalents for English, displaying a high level of quality. However, future work will need to look at how to improve this quality further, especially for other languages, as well as to confirm what exactly the right extent of underspecification is for Deep inputs for surface realisation.

A separate question the field needs to address is to what extent *Shallow* underspecified UD inputs are suitable for surface realisation. More specifically, whether it is reasonable to expect other, content-determining, modules in an NLG system to generate such inputs. Finally, an overlapping question is whether inputs of either the Shallow or Deep type provide information that is sufficient for generating fully realised sentences, and if not, how such inputs can be enriched to provide it.

Acknowledgments

The work presented here has been supported in part by three projects funded by the European Commission: V4Design (H2020-779962-RIA), TENSOR (H2020-700024-RIA), and beAWARE (H2020-700475-RIA).

References

Miguel Ballesteros, Bernd Bohnet, Simon Mille, and Leo Wanner. 2015. Data-driven deep-syntactic dependency parsing. *Natural Language Engineering*, pages 1–36.

John Bateman. 1997. Enabling technology for multilingual natural language generation: The KPML development environment. *Natural Language Engineering Journal*, 3(1):15–55.

Anja Belz and Eric Kow. 2010. The GREC challenges 2010: Overview and evaluation results. In *Proceedings of the 6th International Natural Language Generation Conference*, INLG'10, pages 219–229, Stroudsburg, PA, USA. Association for Computational Linguistics.

Anja Belz, Michael White, Dominic Espinosa, Eric Kow, Deirdre Hogan, and Amanda Stent. 2011. The first surface realisation shared task: Overview and evaluation results. In *Proceedings of the 13th European Workshop on Natural Language Generation*, ENLG '11, pages 217–226, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alena Böhmová, Silvie Cinková, and Eva Hajičová. 2005. A manual for tectogrammatical layer annotation of the Prague Dependency Treebank (English translation). Technical report, ÚFAL MFF UK, Prague, Czech Republic.

Bernd Bohnet and Leo Wanner. 2010. Open soucre graph transducer interpreter and grammar development environment. Valletta, Malta.

Aoife Cahill and Josef van Genabith. 2006. Robust PCFG-based generation using automatically acquired LFG approximations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1033–1040. Association for Computational Linguistics.

M. Elhadad and J. Robin. 1996. An overview of SURGE: a reusable comprehensive syntactic realization component. Technical report, Ben Gurion University in the Negev.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating training corpora for micro-planners. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada. Association for Computational Linguistics.

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

Irene Langkilde-Geary. 2002. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proceedings of the International Natural Language Generation Conference (INLG'02)*.

Percy Liang, Michael I Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 91–99. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal stanford dependencies: A cross-linguistic typology. pages 4585–4592, Reykjavik, Iceland. European Language Resources Association (ELRA). ACL Anthology Identifier: L14-1045.

Jonathan May and Jay Priyadarshi. 2017. Semeval-2017 task 9: Abstract meaning representation parsing and generation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 534–543, Vancouver, Canada. Association for Computational Linguistics.

Adam Meyers, R. Reeves, C. Macleod, R. Szekely, V. Zielinska, B. Young, and R. Grishman. 2004. The NomBank project: An interim report. In *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation, Boston, MA, May 2004*, pages 24–31.

Simon Mille, Anja Belz, Bernd Bohnet, Yvette Graham, Emily Pitler, and Leo Wanner. 2018. The First Multilingual Surface Realisation Shared Task (SR'18): Overview and Evaluation Results. In *Proceedings of the 1st Workshop on Multilingual Surface Realisation (MSR), 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1–12, Melbourne, Australia.

Simon Mille, Roberto Carlini, Ivan Latorre, and Leo Wanner. 2017. UPF at EPE 2017: Transduction-based deep analysis. In *Shared Task on Extrinsic Parser Evaluation (EPE 2017)*, pages 80–88, Pisa, Italy.

Simon Mille and Leo Wanner. 2015. Towards large-coverage detailed lexical resources for data-to-text generation. In *Proceedings of the First International Workshop on Data-to-text Generation*, Edinburgh, Scotland.

Hiroko Nakanishi, Ysuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic models for disambiguation of an HPSG-based chart generator. In *Proceedings of the International Workshop on Parsing Technologies*.

Jekaterina Novikova, Ondrej Dušek, and Verena Rieser. 2017. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Saarbrücken, Germany. ArXiv:1706.09254.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–105.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the International Conference on Computational Linguistics and the Annual Meeting of the Association for Computational Linguistics (COLING-ACL'08)*.

Michael White and Rajakrishnan Rajkumar. 2009. Perceptron reranking for CCG realization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'09)*.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajic, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, et al. 2017. Conll 2017 shared task: multilingual parsing from raw text to universal dependencies. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19.

Huayan Zhong and Amanda Stent. 2005. Building surface realizers automatically from corpora using general-purpose tools. In *Proceedings of the Workshop on Using Corpora for Natural Language Generation (UCNLG)*.