FORGe at SemEval-2017 Task 9: Deep sentence generation based on a sequence of graph transducers

Simon Mille¹, Roberto Carlini¹, Alicia Burga¹, Leo Wanner^{1,2}

¹ Universitat Pompeu Fabra, Roc Boronat 138, 08018 Barcelona, Spain ²Institució Catalana de Recerca i Estudis Avançats (ICREA),

Lluis Companys 23, 08010 Barcelona, Spain

firstname.lastname@upf.edu

Abstract

We present the contribution of Universitat Pompeu Fabra's NLP group to the Sem-Eval Task 9.2 (AMR-to-English Generation). The proposed generation pipeline comprises: (i) a series of rule-based graphtransducers for the syntacticization of the input graphs and the resolution of morphological agreements, and (ii) an off-theshelf statistical linearization component.

1 Setup of the system

The generator we presented for Task 9.2 of Sem-Eval is a pipeline of graph transducers called *Fabra Open Rule-based Generator* (FORGe).¹ It is built upon work presented, e.g., in (Bohnet, 2006; Wanner et al., 2010). It can be also considered an extended rule-based version of (Ballesteros et al., 2015). The current generator has been mainly developed on the dependency Penn Treebank (Johansson and Nugues, 2007) automatically converted to predicate-argument structures, and adapted to the AMR inputs using SemEval's training and evaluation sets.

1.1 Overview of the pipeline

The core of the generator is rule-based: graph transduction grammars convert, in several steps, abstract AMRs into syntactic structures that contain all the morphological features needed to retrieve the final forms of all the words. The syntactic structures are then linearized with an off-the-shelf tool, and finally the final forms of the words are retrieved. Our generator follows the theoretical model of the Meaning-Text Theory (Mel'čuk, 1988); the names of the intermediate layers used in Table 1 come from the MTT terminology.

| | Step | Layer _{mtt} | #rul. |
|---|------------------------------|-----------------------|-------|
| 0 | Conversion of AMRs format | ConS | N/A |
| | into CoNLL'09 format | | |
| 1 | Mapping of AMRs onto | SemS | 190 |
| | predicate-argument graphs | | |
| 2 | Assignment of parts | SemS _{pos} | 96 |
| | of speech | | |
| 3 | Derivation of deep syntactic | DSyntS | 267 |
| | structure | | |
| 4 | Introduction of function | SSyntS | 294 |
| | words | | |
| 5 | Resolution of agreements | DMorphS | 85 |
| 6 | Linearization | SMorphS | N/A |
| 7 | Retrieval of surface forms | Text | 1 |
| 8 | Post-processing | Text _{final} | 4 |

Table 1: Overview of the AMR-to-text pipeline.

1.2 Input format conversion

Since our generator cannot read the provided format, we converted the input AMRs to the CoNLL'09 format (Hajič et al., 2009). We assume that each sentence in the original file has two components: a three-line comment with some metadata (id, date, original sentence, etc.) and the AMR tree. In order to map the AMR tree to CoNLL, we assume that: (i) each node of the tree will be defined as either (a) slash-separated variable-value pair, (b) only the variable name, or (c) only the value, and (ii) each branch will be defined by a relation name preceded by a colon.

2 Generation from AMRs

There are 932 activated graph-transduction rules in the pipeline; Steps 0 and 1 are AMR-specific, while in the rest of the grammars, only one rule is AMR-specific rule (at Step 3).

2.1 Mapping of AMRs onto predicate-argument graphs

The mapping produces graphs that contain linguistic information only, which includes meaning bearing units and the following types of roles:

¹A slightly updated version of the submitted system can be found at https://www.upf.edu/web/taln/ resources

Arguments and coordinations (ARG0, ARG1, ..., ARGn), relations coming from AMR non-core and prepositional roles (NonCore), and underspecified relations (Elaboration). Information that does not need to be generated in the final sentence is excluded: ontological information such as types of entities (e.g., "person", "date-entity") and relations (e.g., "has-rel-role", "has-org-role", "month"); meta-information such as the origin of a label (e.g., "Wiki"); etc. For this purpose, we remove the corresponding nodes and relations and connect the remaining nodes. Furthermore, the AMR core roles are maintained, while all other roles (non-core, prepositional, coordinations) are mapped to one single role or to a subgraph that contains one node and two edges, as, e.g., in the following example:

$$N_1 N_2 \rightarrow N_1 N_{purpose} N_2$$

Since AMRs are provided in a tree format, there are some reversed argumental relations (ARG0-of, ARG1-of, etc.). For the sake of a consistent treatment of all argumental relations, these relations are inverted back in our setting and the "-of" extension is removed.²

2.2 Assignment of parts of speech

Before building the syntactic structures, we assign parts of speech to each node of the structure. This tagging is very rudimentary and slow: rules check in a lexicon if a node label matches an entry and retrieve the part of speech from that entry. Since more than one part of speech is often possible for one string, rules consult the context in the graph, if necessary, in order to take a decision.

2.3 Derivation of the deep syntactic structure

This transduction performs a top-down recursive syntacticization of the semantic graph. It looks for the syntactic root of the sentence, and from there for its syntactic dependent(s), for the dependent(s) of the dependent(s), and so on.

The rules are organized in a cluster of three grammars. The purpose of the first two grammars is to identify the root of a syntactic tree in case the original input structure does not contain one.³

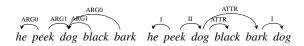
Obviously, verbs are the best candidates for being the root; we pay special attention to the number and complexity of dependents that a verb has (a "heavier" verb is more likely to be a root), and to the fact that a verb is not on the ARG2 side of a node that comes from a non-core relation, since this configuration makes syntacticization less natural in many cases. Consider the following AMR graph:



In this example, if the first "go" is chosen as the root, the syntactic tree can unfold as "I go if you go". But if the second "go", which is on the ARG2 side of the "have-condition" non-core node, is selected as the root, the generator is currently unable to build a sentence, since it should be able to invert the "if" and has no such information.⁴

For this task, the main challenge is to produce a well-formed tree that covers as much of the input graph as possible, while avoiding the possible dependency conflicts. One major issue to solve is that one element can (legitimately) be the argument of several predicates, whereas (i) a syntactic structure is a tree-like structure in which every node can have only one governor, (ii) some predicates *need* their argument(s) in syntax, and (iii) some predicates cannot be realized with their argument(s). In other words, each argument has to be assigned to the correct governor in syntax, and duplicated when needed. The rules keep track of whether a node has already been used and what it has been used for.⁵

In the following example, "peek" is chosen as the root, "dog" is its dependent, and the other predicates of which "dog" was an argument become its syntactic dependents. Furthermore, "dog" is duplicated in order to create a relative clause (Left: predicate-argument; Right: Deep-Syntax):



nodes are removed (see 2.1), it is possible that the original root is missing, and the system needs to find a substitute. More generally, we want our system to be able to generate without any information about syntactic structure.

 $^{^{2}}$ We are conscious that this transformation makes the task more challenging. For instance, it sometimes creates cycles in the graphs. Specific rules address this issue.

³Given that AMRs are provided in tree format, and the root of this tree is the main node (the root) of the sentence, this step can seem redundant. But since during Step 1 some

⁴Even for a human it is challenging to generate a natural sentence; maybe something like "The fact that you go would be the condition for me to go" would be acceptable.

⁵For more details on the deep-syntactic structures and their relation with surface-syntactic structures, see (Balles-teros et al., 2016).

2.4 Introduction of function words

The next step towards the realization of the sentence is the introduction of all idiosyncratic words and of a fine-grained (surface-)syntactic structure that gives enough information for linearizing and resolving agreements between the different words. For this task, we use a valency (subcategorization) lexicon built automatically from PropBank (Kingsbury and Palmer, 2002) and NomBank (Meyers et al., 2004); see (Mille and Wanner, 2015). For instance, the entry corresponding to "peek" would contain the following information:

peek_VB_01{pos=VB gp={II={prep=at, rel=IOBJ}}}

It indicates that, according to PropBank, the second argument of "peek" needs the preposition "at". Hence, this preposition is introduced in the surface-syntactic structure, as shown in the following example:

be peek at dog the black bark that

AMRs are underspecified in terms of tense, aspect, number, and definiteness. For the task, a past progressive is equally correct as a simple present. Our generator is able to introduce all types of auxiliaries and/or modals, but we set the default verbal form to simple present. By default, nominal number is set to singular and definiteness to indefinite. However, the corresponding rule can take different decisions: in some cases, the presence of a definite determiner is usually more adequate-for instance, if the noun is modified by a relative clause (cf. the above example). During this transduction, anaphora are resolved, and personal pronouns are introduced in the tree (this includes possessive, relative and personal pronouns). Some syntactic post-processing rules fix ill-formed structures when possible.

2.5 Resolution of morpho-syntactic agreements

Every word must be assigned all necessary morphological information; some information comes from the deeper strata of the pipeline (as, e.g., verbal tense or finiteness), but some features come from some other elements of the tree. In English, for instance, verbs get their person and number from their subject. In order to resolve these agreements, the rules for this transduction check the governor/dependent pairs, together with the syntactic relation that links them together. During this step, parts of speech are set to match the training data used for the linearizer, and question and exclamation marks are introduced.

2.6 Linearization

The surface-syntactic structures are linearized with an off-the-shelf tool used in the first SRST (Belz et al., 2011), a statistical tree linearizer that orders bottom-up each head and its children (Bohnet et al., 2011).

2.7 Retrieval of surface forms

With the morpho-syntactic information at hand for each word, we just need to find the corresponding surface form. We match the <lemma><POS><morpho-syntactic triple features> with an entry of a morphological dictionary and simply replace the triple by In order to build such a the surface form. dictionary, we analyzed a large amount of data⁶ and retrieved all possible combinations of lemma, part of speech and morphological features; e.g., for verbs: peek<VB><GER>=peeking; peek<VB><FIN><PRES><3><SG>=peeks. If several surface forms are found for a combination of features, we keep the most frequent one. The final sentence corresponding to the running example would be He peeks at the black dog that barks.

2.8 Post-processing

A few post-processing rules make the output more readable: the first letter of a sentence is converted to upper case, punctuation signs are added, underscores are replaced by spaces, and spaces before contracted elements ("'s" and "n't") are removed.

3 Results and discussion

| Metric | FORGe | Average |
|-------------|-------|---------|
| Win pct | 43.64 | 36.668 |
| win+tie pct | 57.43 | 54.19 |
| trueskill | 44.9 | 40.664 |
| BLEU | 4.74 | 11.362 |

Table 2: Results of the task

It takes in average about 2 seconds to generate one sentence, 1.7 seconds of which is due to PoS assignment. Table 2 shows the results that our system obtained for the task, compared to the average results of the other systems; the first three metrics are derived from the ranking obtained during the human evaluation; BLEU is the result of an n-gram based automatic evaluation. According to

⁶For this, we used the Open ANC corpus http://www.anc.org/data/oanc/annotations/

the organizers of the task, *trueskill* (highlighted in bold) is considered the "main" metric for this task.

Our system obtained results slightly above average for the human-based evaluation, while the BLEU score is quite low, compared to that of the other systems. We believe that this is due to the fact that we prioritized the quality of the output over coverage: the submitted generator produces an output for 98.8% of the sentences, but the latter only contain 74.3% of the total of nodes contained in the predicate-argument graphs (see Section 2.1). Indeed, at two points of the pipeline, we choose not to generate or to remove content from the trees. First, during Step 3, when the generator is not sure what to do with one predicate-argument relation, it does not generate it. For instance, when the root of a sentence is on the ARG2 side of a node which originates from a non-core relation, the whole subgraph of this node is omitted (see Section 2.3). Second, during Step 4, if a subtree that is likely to be faulty is identified (as, e.g., a conjunction without a complement), it is removed.

Consider for example the following (partial) gold output: Securing Reiss' release has been a diplomatic priority for France, with President Nicolas Sarkozy raising the case with other leaders. Our fallback wholecoverage generator would output diplomacy France prioritize secure like release Clotilde Reiss president Nicolas Sarkozy raise case with onbehalfof person intervene..., while our actual output is France prioritizes to secure the release of Clotilde Reiss. Eventually, choosing the production of readable sentences over their completeness allowed us to get better human ratings. But this, combined to the fact that we always generate simple present tense and singular words, naturally has a negative impact with an n-gram-based metric (that includes a brevity penalty) such as BLEU.7

Finally let us point out that most rules in this system are language-independent. Experiments adapting it to Spanish, Polish and German show that with rich lexicons, the effort related to grammars for obtaining well-formed texts is minimal.⁸

Acknowledgments

The work described in this paper has been partially funded by the European Commission under the contract numbers FP7-ICT-610411, H2020-645012-RIA, H2020-700024-RIA, and H2020-DRS-01-2015.

References

- Miguel Ballesteros, Bernd Bohnet, Simon Mille, and Leo Wanner. 2015. Data-driven sentence generation with non-isomorphic trees. In *Proceedings of NAACL:HLT*. ACL, Denver, CO, pages 387–397.
- Miguel Ballesteros, Bernd Bohnet, Simon Mille, and Leo Wanner. 2016. Data-driven deep-syntactic dependency parsing. *Natural Language Engineering* 22(6):939–974.
- Anja Belz, Mike White, Dominic Espinosa, Eric Kow, Deirdre Hogan, and Amanda Stent. 2011. The first Surface Realisation Shared Task: Overview and evaluation results. In *Proceedings of ENLG*. Nancy, France, pages 217–226.
- Bernd Bohnet. 2006. *Textgenerierung durch Transduktion linguistischer Strukturen*. Ph.D. thesis, University Stuttgart.
- Bernd Bohnet, Simon Mille, Benoît Favre, and Leo Wanner. 2011. StuMaBa: From deep representation to surface. In *Proceedings of ENLG*. Nancy, France, pages 232–235.
- J. Hajič, M. Ciaramita, and R. Johansson et al. 2009. The CoNLL-2009 Shared Task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of CoNLL*. Boulder, CO, USA, pages 1–18.
- Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In *Poceedings of NODALIDA*. Tartu, Estonia, pages 105–112.
- Paul Kingsbury and Martha Palmer. 2002. From Tree-Bank to PropBank. In *Proceedings of LREC*. Las Palmas, Canary Islands, Spain, pages 1989–1993.
- Igor Mel'čuk. 1988. *Dependency Syntax: Theory and Practice*. SUNY Press, Albany.
- Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The NomBank Project: An interim report. In *Proceedings of the Workshop* on Frontiers in Corpus Annotation, (HLT/NAACL). Boston, MA, USA, pages 24–31.
- Simon Mille and Leo Wanner. 2015. Towards largecoverage detailed lexical resources for data-to-text generation. In Proceedings of the First International Workshop on D2T Generation. Edinburgh, Scotland.
- Leo Wanner, Bernd Bohnet, Nadjet Bouayad-Agha, François Lareau, and Daniel Nicklaß. 2010. MAR-QUIS: Generation of user-tailored multilingual air quality bulletins. *Applied Artificial Intelligence* 24(10):914–952.

⁷Note that our fallback generator obtained a BLEU that is only 18% inferior to the score of the submitted output.

 $^{^{8}}$ Currently, for indicative purposes, the approximate count of non language-independent rules is the following: 2% in Step 2, 14% in Step 3, 64% in Step 4, and 13% in Step 5.