Exploiting ontology lexica for generating natural language texts from RDF data

Philipp Cimiano, Janna Lüker, David Nagel, Christina Unger
Semantic Computing Group
Cognitive Interaction Technology – Center of Excellence (CITEC),
Bielefeld University, Germany

Abstract

The increasing amount of machine-readable data available in the context of the Semantic Web creates a need for methods that transform such data into human-comprehensible text. In this paper we develop and evaluate a Natural Language Generation (NLG) system that converts RDF data into natural language text based on an ontology and an associated ontology lexicon. While it follows a classical NLG pipeline, it diverges from most current NLG systems in that it exploits an ontology lexicon in order to capture context-specific lexicalisations of ontology concepts, and combines the use of such a lexicon with the choice of lexical items and syntactic structures based on statistical information extracted from a domain-specific corpus. We apply the developed approach to the cooking domain, providing both an ontology and an ontology lexicon in *lemon* format. Finally, we evaluate fluency and adequacy of the generated recipes with respect to two target audiences: cooking novices and advanced cooks.

1 Introduction

The goal of the Semantic Web is to enrich the current web by a layer of machine-readable and machine-understandable content (Berners-Lee et al., 2001). In recent years, the growth of data published on the web according to Semantic Web formalisms and data models (e.g. RDF(S) and OWL) has been exponential, leading to more than 30 billion RDF triples[1] available as part of the *Linked Open Data* cloud, which contains a wide range of factual knowledge that is very interesting to many applications and for many purposes. However, due to the fact that it is available as RDF, it is not directly accessible to humans. Thus, natural language generation from RDF data has recently become an important topic for research, leading to the development of various systems generating natural language text from knowledge bases (Bouayad-Agha et al., 2012a; Mellish and Sun, 2006; Sun and Mellish, 2007; Wilcock and Jokinen, 2003) as well as corresponding shared tasks (Banik et al., 2012; Bouayad-Agha et al., 2012b).

Natural language generation (NLG) from knowledge bases requires knowledge about how the concepts in the underlying ontology—individuals, classes and relations—are realised linguistically. For this purpose, *lemon*, a lexicon model for ontologies, has been developed (McCrae et al., 2011). One of the use cases of *lemon* is to support natural language generation systems that take as input a knowledge base structured with respect to a given ontology. In this paper, we present a system that relies on *lemon* lexica for selecting suitable lexicalisations of a given concept, showing how ontology lexica can be exploited in a standard generation architecture.

We apply our system to the domain of cooking, generating natural language texts for recipes modeled as RDF data based on a cooking ontology. Our system relies on a large text corpus of cooking recipes that is used to extract frequency information for single terms and n-grams as well as syntactic trees, which are then used in the selection process for lexicalisation and surface realisation. Additionally, we provide a manually created *lemon* lexicon for the underlying ontology that was enriched with inflectional variants derived from

[1] `http://www4.wiwiss.fu-berlin.de/lodcloud/state/`

Proceedings of the 14th European Workshop on Natural Language Generation, pages 10–19,
Sofia, Bulgaria, August 8-9 2013.

Wiktionary. The lexicon also includes contextual information regarding which lexicalisations to prefer depending on the target group, and thereby allows our system to personalize the output to different groups of users. We demonstrate the flexibility of our system by showing that it can be easily tuned to generate recipe descriptions both for novices and for advanced cooks and that this adaptation is clearly recognized by users.

The remainder of this paper is structured as follows. In Section 2 we describe the resources we created and employed, in particular a domain ontology, a corresponding ontology lexicon enriching ontology concepts with lexical information, and a parsed domain corpus. In Section 3 we describe the architecture of the system, in particular the use of a corpus for selecting appropriate syntactic structures and surface realisations of concepts. Then we present the results of an extensive user study in Section 4, compare our approach to related work in Section 5 and finally give an outlook on future work in Section 6.

```
:Marmorkuchen a :Nachspeise;

  :hasStep [ a :Step ;
  :hasStepNumber 7^^xsd:integer ;
  :hasAction     action:mischen ;
  :hasMixType    prop:vermengen ;
  :hasIngredient
       [ a ingredient:Butter ;
         :hasAmount amount:Gramm ;
         :hasValue  "300" ],
       [ a ingredient:Mehl ;
         :hasAmount amount:Gramm ;
         :hasValue  "375" ],
       [ a ingredient:Ei ;
         :hasAmount amount:Stueck
         :hasValue  "5" ] ;
  :hasIndirectIngredient
                  tool:Schuessel ;
  :creates tool:Marmorkuchen_Interim_1
  ] ;

  :hasStep [ a :Step ;
  :hasStepNumber 8^^xsd:integer ;
  :hasAction     action:backen ;
  :isPassive     "true"^^xsd:boolean ;
  :hasTimeUnit   prop:Minute ;
  :hasTimeValue  45.0^^xsd:double ;
  :hasIngredient
      tool:Marmorkuchen_Interim_1 ;
  :hasIndirectIngredient
      tool:Backofen
  ] .
```

Figure 1: An excerpt from the RDF recipe for marble cake.

2 Resources

2.1 Domain ontology and lexicon

In order to be able to model cooking recipes as RDF data, we created a domain ontology in which recipes are modeled comprising the following information (for a similar modeling see (Ribeiro et al., 2006)):

- An indication of the number of people that it serves.
- A set of ingredients used in the recipe.
- An ordered list of steps involving a certain action (e.g. cutting) on a set of ingredients. Each action in turn allows one or many modifiers (e.g. to indicate cutting granularity).
- Interim ingredients that are produced as the result of some step and can be reused later in another step.

An excerpt from the RDF recipe for marble cake is given in Figure 1. It shows two steps, one for mixing the ingredients butter, flour and egg, using a bowl, thereby creating the dough as an interim object, and a subsequent one in which this interim object is being baked in the oven for 45 minutes.

In general, each step comprises:

- A *step number* indicating the order in a list of steps.
- An associated *action* indicating the type of action performed in the step, e.g. *to fold in.*
- One or more *ingredients* used in the action. This is either an ingredient from the ingredient list of the recipe, or an object that was created as a result of some other step.
- A *passivity* flag indicating whether a step does not require an active action by the cook, e.g. *Let the cake cool for 1 hour.*
- Further modifiers such as *mixType* indicating the way in which the ingredients

are mixed (e.g. *beating* or *folding*), temporal modifiers specifying a *time unit* and *time value* (e.g. 45 minutes). These modifiers later affect the grouping of steps and their lexicalisation.

- A flag indicating whether this is a *key step* within the recipe, for example a step that requires particular care and thus should get emphasis in the verbalization, like *Quickly fry the meat!*

Overall, the ontology comprises 54 different action types that we used to manually model 37 recipes. Further, we created a *lemon* lexicon specifying how the different actions and ingredients specified in the ontology are verbalized in German. In total the lexicon contains 1,530 lexical entries, on average 1.13 lexical variants for each ingredient and 1.96 variants for each action.

Figure 2 gives an example entry for the verb *schneiden* (*to cut*), specifying its part of speech, two form variants, the infinitive and the past participle, and a semantic reference to the ontology action of cutting. Figure 3 gives an excerpt from the lexical entry for *tranchieren* (*to carve*), which refers to the same cutting action but is restricted to cases where the ingredient is of type *meat*, modelled using a logical condition that can be issued as a query to the knowledge base. This verb would therefore only be used in the context of technical registers, i.e. with advanced cooks as target group.

After having manually created lexical entries with their base forms, we automatically enrich them with inflectional forms extracted from Wiktionary, as already indicated in Figure 2.

The ontology, the RDF recipes as well as the ontology lexicon can be accessed at `http://www.sc.cit-ec.uni-bielefeld.de/natural-language-generation`.

Although the manual creation of *lemon* lexica is feasible for small domains (and supported by tools such as *lemon source* (McCrae et al., 2012)), it does not scale to larger domains without a significant amount of effort. Therefore corpus-based methods for the semi-automatic creation of ontology lexica are currently developed, see (Walter et al., 2013).

```
:schneiden a lemon:LexicalEntry ;
 lexinfo:partOfSpeech lexinfo:verb ;

 lemon:canonicalForm [
 lemon:writtenRep "schneiden"@de ;
  lexinfo:tense lexinfo:present ;
  lexinfo:mood  lexinfo:infinitive
 ];
 lemon:otherForm [
  lemon:writtenRep "geschnitten"@de ;
  lexinfo:verbFormMood
                lexinfo:participle ;
  lexinfo:aspect lexinfo:perfective
 ];

 lemon:sense
 [ lemon:reference action:schneiden ].
```

Figure 2: Lexical entry for the verb *schneiden*, denoting a cutting action.

```
:tranchieren a lemon:LexicalEntry ;
 lexinfo:partOfSpeech lexinfo:verb ;

 lemon:canonicalForm [
  lemon:writtenRep "tranchieren"@de ];

 lemon:sense
 [ lemon:reference action:schneiden;
   lemon:condition [ lemon:value
    "exists ?x :
     :hasIngredient(?x,?y),
     :Step(?x),
     ingredient:Fleisch(?y)" ];
   lemon:context
     isocat:technicalRegister ] .
```

Figure 3: Lexical entry for the verb *tranchieren*, denoting a cutting action restricted to meat and marked as a technical term.

2.2 Domain corpus

In order to build a domain corpus, we crawled the recipe collection website `http://www.chefkoch.de`, which at that point contained more than 215 000 recipes with a total amount of 1.9 million sentences. We extracted the recipe text as well as the list of ingredients and the specified level of difficulty – *easy*, *normal* and *complicated*.

The extracted text was tokenized using the unsupervised method described by Schmid (Schmid, 2000), and for each recipe an n-gram index (considering 2, 3 and 4-grams) for both the recipe text and the ingredient list was constructed. Furthermore, 65 000 sentences were parsed using the Stanford parser, trained on

the German TIGER corpus, also enriching the training data of the parser with fragments derived from the ontology lexicon in order to ensure that the lexical entries in the ontology lexicon are actually covered. This resulted in 20 000 different phrase structure trees where the leafs were replaced by lists of all terms occurring at that position in the parse tree. Both trees and leaf terms were stored together with the number of their occurrences. Leaf terms were additionally annotated with lexical senses by comparing them to the already created lexical entries and thus connecting them to ontology concepts.

3 System architecture

Our system implements a classical NLG pipeline comprising the following three steps (Reiter and Dale, 2000):

- Document planning
- Microplanning
- Surface realisation

Document planning in our case is quite straightforward as the recipes already comprise exactly the information that needs to be verbalized. In the following we present the two remaining steps in more detail, followed by a brief description of how the text generation is parametrized with respect to the target group (novices or experts).

3.1 Microplanning

Following Reiter & Dale (Reiter and Dale, 2000), microplanning comprises three steps: aggregation, referring expression generation, and lexicalisation.

Aggregation Aggregation serves to collapse information using grouping rules in order to avoid redundancies and repetitions. In our case, the main goal of aggregation is to group steps of recipes, deciding which steps should be verbalized within the same sentences and which ones should be separated, based on the following hand-crafted rules:

- Steps are grouped if
  - they have the same step number, or
  - the actions associated with the steps are the same, or
  - the same ingredient is processed in subsequent actions, e.g. peeling and chopping onions.
- Steps that are marked as *important* in the ontology can only be grouped with other important steps.
- If the grouping of steps would result in too many ingredients to still form a readable sentence, the steps are not grouped. Currently we consider more than six ingredients to be too many, as there are hardly any trees in the corpus that could generate corresponding sentences.
- If there is a big enough time difference between two steps, as e.g. between baking a cake for 60 minutes and then decorating it, the steps are not grouped.

Each of these rules contributes to a numerical value indicating the probability with which steps will be grouped. The use of the rules is also controlled by a system parameter λ_{length} that can be set to a value between 0 and 1, where 0 gives a strong preference to short sentences, while 1 always favors longer sentences.

Referring expression generation The generation of referring expressions is also rule-based and mainly concerns ingredients, as actions are commonly verbalized as verbs and tools (such as bowls and the oven) usually do not re-occur often enough. In deciding whether to generate a pronoun, the following rule is used: A re-occurring ingredient is replaced by a pronoun if there is no other ingredient mentioned in the previous sentence that has the same number and gender. A system parameter $\lambda_{pronoun}$ can be set to determine the relative frequency of pronouns to be generated.

If an ingredient is not replaced by a pronoun, then one of the following expressions is generated:

- A full noun phrase based on the verbalization given in the ontology lexicon, e.g. *two eggs*.
- A definite expression describing a super-category of the given ingredient. The super-category is extracted from the ontology and its verbalization from the on-

tology lexicon. For instance, if the ingredient in question is *pork*, the expression *meat* would be generated.

- A zero anaphora, i.e. an empty referring expression, as in *Bake for 60 minutes* or *Simmer until done.*

The use of those variants is regulated by a system parameter $\lambda_{pronoun}$, where a high value forces the use of abstract expressions and zero anaphora, while a low value prefers the use of exact ingredient names. In future work the decision of which referring expression to use should be decided on the basis of general principles, such as uniqueness of the referent, avoidance of unnecessary and inappropriate modifiers, brevity, and preference for simple lexical items, see, e.g., (Reiter and Dale, 1992).

An exception to the above rules are interim ingredients, whose realisation is determined as follows. If there is a lexical entry for the interim, it is used for verbalization. If there is no lexical entry, then the name of the main ingredient used in the creation of the interim is used. Furthermore, we define and exploit manually specified meaning postulates to create names for specific, common interims. For example *dough* is used if the interim is generated from *flour* and at least one of the ingredients *butter*, *sugar*, *egg* or *backing powder*.

Lexicalisation In order to lexicalise actions and ingredients, the ontology lexicon is consulted. Especially for actions, the lexicon contains several lexical variants, usually accompanied by a restriction that specifies the context in which the lexicalisation is appropriate. For example the action *to cut* can be lexicalised in German as *hacken* (*to chop*) if the specified granularity is *rough*, as *blättrig schneiden* (*to thinly slice*) if the specified granularity is *fine*, or *tranchieren* (*to carve*) in case the ingredient is of type *meat*.

The conditions under which a lexicalisation can be used felicitously are given in the lexicon as logical expressions, as exemplified in Figure 3 above, which are translated into SPARQL queries that can be used to check whether the condition is satisfied with respect to the recipe database.

In addition, we rely on statistics derived from our domain corpus in order to choose a lexicalisation in case the conditions of more than one lexical variant are fulfilled, by preferring terms and term combinations with a higher frequency in the domain corpus. Again, the system implements a parameter, $\lambda_{variance}$, that regulates how much overall lexical variability is desired. This, however, should be used with care, as choosing variants that are less frequent in the corpus could easily lead to strange or inappropriate verbalizations.

3.2 Surface realisation

The input to the surface realisation component is a list of concepts (spanning one or more recipe steps) together with appropriate lexicalisations as selected by the lexicalisation component. The task of the surface realiser then is to find an appropriate syntactic tree from the parsed corpus that can be used to realise the involved concepts. An example of such a parse tree with annotated leaf probabilities is shown in Figure 4.

All trees retrieved from the index are weighted to identify the best fitting tree combining the following measures: i) the normalized probability of the syntax tree in the domain corpus, ii) a comparison of the part-of-speech tag, synonyms and the lexical sense of a given lexicalisation with those of the terms in the retrieved tree, iii) the node distances of related words inside each tree, and iv) an n-gram score for each resulting sentence. These scores are added up and weighted w.r.t. the size of n, such that, for example, 4-grams have more influence on the score than 3-grams. Also, sentences with unbalanced measure, i.e. that score very well w.r.t. one measure but very poorly w.r.t. another one, are penalized.

3.3 Personalization

On the basis of conditions on the context of use provided in the ontology lexicon, it is possible to distinguish lexicalisations that are suitable for experts from lexical variants that are suitable for novices. Thus, texts can be generated either containing a high amount of technical terms, in case the user has a high proficiency level, or avoiding technical terms at all, in case the user is a novice. Furthermore, the complexity of texts can be varied by adjusting the

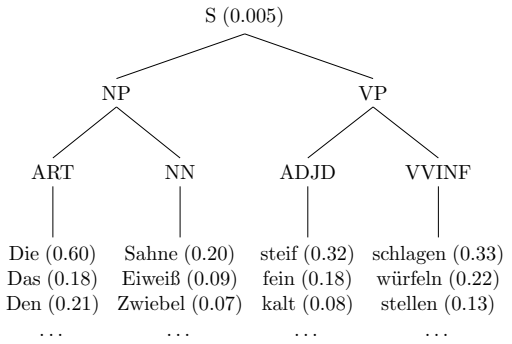

Figure 4: Example of a parse tree extracted from the corpus, annotated with leaf probabilities

sentence length and the number of adjectives used. We used this as an additional parameter $\lambda_{context}$ for tailoring texts to their target group, preferring complex structures in expert texts and simple structures in texts for novices. The influence of this parameter is tested as part of the user study described in the next section.

Personalization thus has been implemented at the level of microplanning. In addition, personalization is possible on the level of text planning. For example, experts often require less detailed descriptions of actions, such that they can be summarized in one step, while they need to be broken down into several steps for beginners. This will be subject of future work.

4 Evaluation

The system was evaluated in an online study with 93 participants—mainly students recruited via email or Facebook. The majority of the participants (70%) were between 18 and 34 years old; the native tongue of almost all participants (95%) was German. About half of the participants regarded themselves as novices, while the other half regarded themselves as advanced cooks.

For each participant, 20 recipes were randomly selected and split into two groups. For ten recipes, test subjects were asked to rate the fluency and adequacy of the automatically generated text along the categories *very good*, *good*, *sufficient* and *insufficient*. The other ten recipes were used to compare the effect of parameters of the generation system and thus were presented in two different versions, varying the sentence length and complexity as well as the level of proficiency. Participants were asked to rate texts as being appropriate *for novices* or *for advanced cooks*.

The parameters that were varied in our experimental setting are the following:

- $\lambda_{context}$: The context of the used terms, in particular *novice* or *advanced*.
- $\lambda_{pronoun}$: Amount of proper nouns, where a high value prefers pronouns over proper nouns, while a low value generates only proper nouns.
- $\lambda_{variance}$: Amount of repetitions, where low values lead to always using the same term, whereas high values lead to fewer repetitions.
- λ_{length}: Length of the created sentences, where a low value creates short sentences, and high values merge short sentences into longer ones.

The values of these parameters that were used in the different configurations are summarized in Table 1. The parameter $\lambda_{pronoun}$ is not varied but set to a fixed value that yields a satisfactory generation of referring expressions, as texts with smaller or higher values tend to sound artificial or incomprehensible.

	$\lambda_{context}$	$\lambda_{pronoun}$	$\lambda_{variance}$	λ_{length}
Standard	*novice*	0.5	0.5	0.5
Novice vs	*novice*	0.5	0.5	0.3
Advanced	*advanced*	0.5	0.5	0.7
Simple vs	*novice*	0.5	0.0	0.3
Complex	*novice*	0.5	1.0	0.7

Table 1: The used parameter sets

Fluency and adequacy of the generated texts Each participant was asked to rate fluency and adequacy of ten automatically generated texts. The results are given in Figures 5 and 6. The fluency of the majority of generated texts (85.8%) were perceived as *very good* or *good*, whereas only 1% of the generated texts were rated as *insufficient*. Similarly, the adequacy of 92.5% of the generated texts were rated as *very good* or *good*, and again only 1% of the generated texts were rated as *insufficient*. There was no significant difference between judgments of novices and experts; neither did the category of the recipe (main or

side dish, dessert, etc.) have any influence. Overall, these results clearly show that the quality of the texts generated by our system is high.

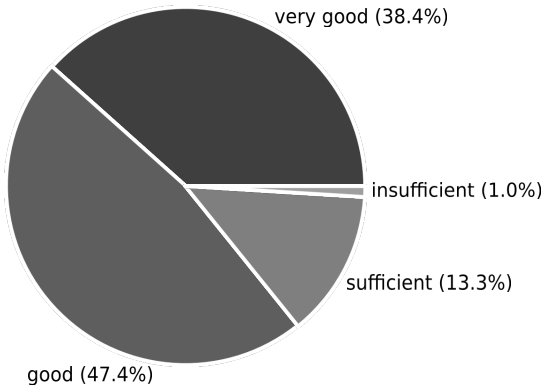

Figure 5: Results for text fluency

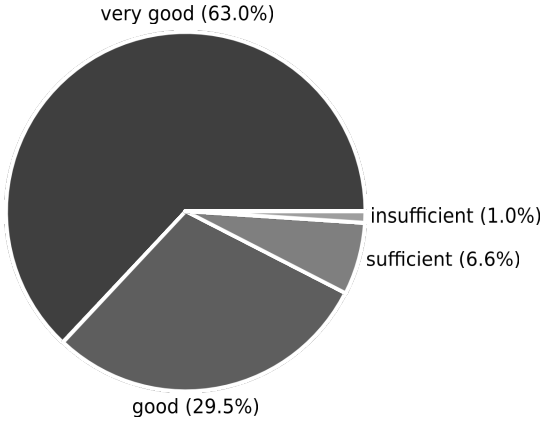

Figure 6: Results for text adequacy

Error analysis The most frequent errors found in the generated texts can be grouped into the following categories:

- **Content (39.4%):** Errors in document planning (e.g. due to the ontology missing details about tools, such as for cutting cookies, or the recipe missing information about the amount of ingredients) or aggregation (e.g. sentences with highly related content were not aggregated), as well as sentence repetitions.
- **Language (29.4%):** Errors in the referring expression generation or lexicalisation steps (e.g. wrong use of function words like *as well*) and grammar errors (e.g. wrong use of definite or indefinite determiners).
- **Other (31.3%):** Some users specified that they would prefer another ordering of the involved steps, or that they lack knowledge of particular terms. Also short sentences with exclamation marks are often perceived as impolite.

Influence of parameter settings We set up the following hypotheses, validating them by means of a χ^2-test by comparing answers across two conditions corresponding to different parameter settings. We regarded a p-value of 0.05 as sufficient to reject the corresponding null hypothesis.

H1 Users prefer longer sentences: Rejecting the null hypothesis that users rate texts with longer sentences and texts with shorter sentences in the same way (p-value: $3 * 10^{-5}$).

H2 Texts for professionals are regarded as not suitable for novices: Rejecting the null hypothesis that texts generated for professionals are regarded as many times as suitable for novices as for professionals (p-value: $2 * 10^{-7}$).

H3 Beginners prefer texts generated for novices: The null hypothesis that novices equally prefer texts targeted to novices and texts targeted to experts could not be rejected.

H4 Advanced cooks prefer texts generated for advanced cooks: Rejecting the null hypothesis that advanced cooks equally prefer texts targeted to novices and texts targeted to experts (p-value: 0.0005).

The confirmation of H1 shows that users perceive a difference in sentence length and prefer texts with longer sentences, probably due to perceived higher fluency. The confirmation of H2 and H4, on the other hand, corroborates the successful adaptation of the generated texts to specific target groups, showing

that texts generated for professionals are indeed perceived as being generated for professionals, and that such texts are preferred by advanced cooks. The rejection of H3 might be caused by the fact that recipes for advanced cooks include some but actually not many technical terms and are therefore also comprehensible for novices.

5 Related work

There have been different approaches to natural language generation, ranging from template-based to statistical architectures. While early NLG systems were mainly based on manually created rules (Bourbeau et al., 1990; Reiter et al., 1992), later approaches started applying statistical methods to the subtasks involved in generation (Belz, 2005), focusing on scalability and easy portability and often relying on overgeneration and subsequent ranking of generation possibilities. Personalization has been a concern in both strands of research. PEBA-II (Milosavljevic et al., 1996), for example, generates target-group-specific texts for novice and experts users from taxonomical information, relying on a phrasal lexicon that is similar in spirit to our ontology lexicon. Statistical approaches such as (Isard et al., 2006), on the other hand, use text corpora to generate personalized texts.

Our approach is hybrid in the sense that it enriches a classical rule-based approach with statistical data in the microplanning and realisation steps, thus being comparable to systems like HALogen (Langkilde and Knight, 1998) and *p*CRU (Belz, 2008). The main difference is that it uses Semantic Web data as base.

Since the emergence of the Semantic Web there has been a strong interest in NLG from Semantic Web data, especially for providing users with natural language access to structured data. Work in this area comprises verbalization of ontologies as well as RDF knowledge bases; for an overview see (Bouayad-Agha et al., to appear). Of particular interest in the context of our work is NaturalOWL (Galanis and Androutsopoulos, 2007), a system that produces descriptions of entities and classes relying on linguistic annotations of domain data in RDF format, similar to our exploitation of ontology lexica. We thus share with NaturalOWL the use of linguistic resources encoded using standard Semantic Web formats. The main difference is that the annotations used by NaturalOWL comprise not only lexical information but also microplans for sentence planning, which in our case are derived statistically and represented outside the lexicon. Separating lexical information and sentence plans makes it easier to use the same lexicon for generating different forms of texts, either with respect to specific target groups or stylistic variants.

6 Conclusion and future work

We have presented a principled natural language generation architecture that follows a classical NLG architecture but exploits an ontology lexicon as well as statistical information derived from a domain corpus in the lexicalisation and surface realisation steps. The system has been implemented and adapted to the task of generating cooking recipe texts on the basis of RDF representations of recipes. In an evaluation with 93 participants we have shown that the system is indeed effective and generates natural language texts that are perceived as fluent and adequate. A particular feature of the system is that it can personalize the generation to particular target groups, in our case cooking novices and advanced cooks. The information about which lexicalisation to prefer depending on the target group is included in the ontology lexicon. In fact, the ontology lexicon is the main driver of the generation process, as it also guides the search for appropriate parse trees. It thus is a central and crucial component of the architecture.

While the system has been adapted to the particulars of the cooking domain, especially concerning the generation of referring expressions, the architecture of the system is fairly general and in principle the system could be adapted to any domain by replacing the ontology, the corresponding ontology lexicon and by providing a suitable domain corpus. This flexibility is in our view a clear strength of our system architecture.

A further characteristic of our system is the consistent use of standards, i.e. OWL for the ontology, RDF for the actual data to be

verbalized, SPARQL for modelling contextual conditions under which a certain lexicalisation is to be used, and the *lemon* format for the representation of the lexicon-ontology interface. One important goal for future work will be to clearly understand which knowledge an ontology lexicon has to include in order to optimally support NLG. To this end, we intend to test the system on other domains, and at the same time invite other researchers to test their systems on our data, available at `http://www.sc.cit-ec.uni-bielefeld.de/natural-language-generation`.

Acknowledgment

This work was partially funded within the EU project PortDial (FP7-296170).

References

E. Banik, C. Gardent, D. Scott, N. Dinesh, and F. Liang. 2012. KBGen: text generation from knowledge bases as a new shared task. In *Proc. Seventh International Natural Language Generation Conference (INLG 2012)*, pages 141–145.

A. Belz. 2005. Statistical generation: Three methods compared and evaluated. In *Proc. 10th European Workshop on Natural Language Generation (ENLG '05)*, pages 15–23.

A. Belz. 2008. Automatic generation of weather forecast texts using comprehensive probabilistic generation-space models. *Natural Language Engineering*, 14(4):431–455.

T. Berners-Lee, J. Hendler, and O. Lassila. 2001. The Semantic Web. *Scientific American Magazine*.

N. Bouayad-Agha, G. Casamayor, S. Mille, M. Rospocher, H. Saggion, L. Serafini, and L. Wanner. 2012a. From ontology to NL: Generation of multilingual user-oriented environmental reports. In *Proc. 17th International Conference on Applications of Natural Language Processing to Information Systems (NLDB 2012)*, pages 216–221.

N. Bouayad-Agha, G. Casamayor, L. Wanner, and C. Mellish. 2012b. Content selection from Semantic Web data. In *Proc. Seventh International Natural Language Generation Conference (INLG 2012)*, pages 146–149.

N. Bouayad-Agha, G. Casamayor, and L. Wanner. to appear. Natural Language Generation in the context of the Semantic Web. *Semantic Web Journal*.

L. Bourbeau, D. Carcagno, E. Goldberg, R. Kittredge, and A. Polguère. 1990. Bilingual generation of weather forecasts in an operations environment. In *Proc. 13th International Conference on Computational Linguistics (COLING 1990)*, pages 318–320.

D. Galanis and I. Androutsopoulos. 2007. Generating multilingual descriptions from linguistically annotated OWL ontologies: the NaturalOWL system. In *Proc. 11th European Workshop on Natural Language Generation (ENLG '07)*, pages 143–146.

A. Isard, C. Brockmann, and J. Oberlander. 2006. Individuality and alignment in generated dialogues. In *Proc. Fourth International Natural Language Generation Conference (INLG 2006)*, pages 25–32.

I. Langkilde and K. Knight. 1998. Generation that exploits corpus-based statistical knowledge. In *Proc. 17th International Conference on Computational Linguistics (COLING '98)*, pages 704–710.

J. McCrae, D. Spohr, and P. Cimiano. 2011. Linking lexical resources and ontologies on the semantic web with lemon. In *Proc. 8th Extended Semantic Web Conference on The Semantic Web: Research and Applications (ESWC 2011)*, pages 245–259.

J. McCrae, E. Montiel-Ponsoda, and P. Cimiano. 2012. Collaborative semantic editing of linked data lexica. In *Proceedings of the 2012 International Conference on Language Resource and Evaluation*.

C. Mellish and X. Sun. 2006. The Semantic Web as a linguistic resource: Opportunities for natural language generation. *Knowl.-Based Syst.*, 19(5):298–303.

M. Milosavljevic, A. Tulloch, and R. Dale. 1996. Text generation in a dynamic hypertext environment. In *Proc. 19th Australian Computer Science Conference*, pages 417–426.

E. Reiter and R. Dale. 1992. A fast algorithm for the generation of referring expressions.

E. Reiter and R. Dale. 2000. *Building natural language generation systems*. Cambridge University Press.

E. Reiter, C. Mellish, and J. Levine. 1992. Automatic generation of on-line documentation in the IDAS project. In *Proc. Third Conference on Applied Natural Language Processing (ANLP)*, pages 64–71.

R. Ribeiro, F. Batista, J.P. Pardal, N.J. Mamede, and H.S. Pinto. 2006. Cooking an ontology. In *Proceedings of the 12th international conference on Artificial Intelligence: methodology, Systems,*

and Applications, AIMSA'06, pages 213–221. Springer.

Helmut Schmid. 2000. Unsupervised learning of period disambiguation for tokenisation. Technical report, IMS-CL, University of Stuttgart.

X. Sun and C. Mellish. 2007. An experiment on "free generation" from single RDF triples. In *Proc. 11th European Workshop on Natural Language Generation (ENLG '07)*, pages 105–108.

S. Walter, C. Unger, and P. Cimiano. 2013. A corpus-based approach for the induction of ontology lexica. In *Proceedings of the 18th International Conference on the Application of Natural Language to Information Systems (NLDB 2013)*.

G. Wilcock and K. Jokinen. 2003. Generating responses and explanations from RDF/XML and DAML+OIL. In *Knowledge and Reasoning in Practical Dialogue Systems, IJCAI 2003 Workshop*, pages 58–63.