Template-based multilingual football reports generation using Wikidata as a knowledge base

Lorenzo Gatti
Human Media Interaction
University of Twente
Enschede, The Netherlands
l.gatti@utwente.nl

Chris van der Lee
Tilburg center for Cognition and Communication
Tilburg University
Tillburg, The Netherlands
c.vdrlee@tilburguniversity.edu

Mariët Theune
Human Media Interaction
University of Twente
Enschede, The Netherlands
m.theune@utwente.nl

Abstract

This paper presents a new version of a football reports generation system called PASS. The original version generated Dutch text and relied on a limited hand-crafted knowledge base. We describe how, in a short amount of time, we extended PASS to produce English texts, exploiting machine translation and Wikidata as a large-scale source of multilingual knowledge.

1 Introduction

One of the advantages of natural language generation is the possibility of producing multilingual texts, e.g. for describing the same data in multiple languages that are familiar to different user groups. Within the extensive literature on NLG, however, relatively few systems have dealt with the problem of multilingual generation, despite the fact that NLG systems for many languages have been developed. It is telling that even the recent NLG survey by Gatt and Krahmer (2018) does not mention it directly.

One of the problems that hinder multilingual NLG research is the additional work required for generating text in different languages. In complex, grammar-based surface realizers such as KPML (Bateman, 1997) and FUF/SURGE (Elhadad and Robin, 1996), adding a new language requires developing a new grammar, a difficult and time-consuming step. Even statistical approaches to NLG, such as those used by OpenCCG (White and Rajkumar, 2009) or recent multilingual systems (Dušek and Jurčíček, 2013; Mille et al., 2016), require some kind of multilingual knowledge base, or at least a large multilingual lexicon. For SimpleNLG, the simplest surface realisation system currently available, it took five months to create a bilingual English-French version (Vaudry and Lapalme, 2013).

Multilingual template-based systems often do not suffer from this problem. The domain is usually limited, such as in the case of reports on weather (Chevreau et al., 1999) or air quality (Busemann and Horacek, 1997), hence a small lexicon is sufficient; in many cases, even the knowledge base is unlikely to periodically need major revisions. The latter is not the case for sports report generation, however: in football, the composition of each team can vary from season to season, a new team can get promoted into a league from a lower one, or a new player might have a nickname in a language that is not used in another language (e.g. Gabriel Batistuta was known in Italy as "Batigol" or "Re Leone", while Spanish speakers nicknamed him "El Ángel Gabriel").

In this paper we present our efforts in dealing with these issues, that we have faced while adding a new language to PASS (van der Lee et al., 2017), a template-based generation system that produces football match reports in Dutch.

Our contribution is twofold. First, we added the possibility to generate English reports, while limiting the changes to the current architecture and without losing support for Dutch. Second, instead of using a hand-crafted local knowledge base, we have integrated the system with the knowledge present in Wikidata. Since the website where PASS collects game statistics reports data about most national leagues, this extension makes it possible to exploit the full potential of the system. That is, generating reports of consistent quality for

Proceedings of The 11th International Natural Language Generation Conference, pages 183–188,
Tilburg, The Netherlands, November 5-8, 2018.

```
<xml>
<MatchData>
 <Highlights>
  <League>Eerste Divisie</League>
  <StartDate>October 30, 2015</StartDate>
  <StartTime>19:00</StartTime>
  <Stadium>Mitsubishi Forklift-Stadion</Stadium>
  <City>Almere</City>
  <Attendees>906</Attendees>
  <Home>
   <Team>Almere City</Team>
   <GoalScorersList>
    <Goal GoalId="1" Minute="45" OwnGoal="n">
     K.Tadmine</Goal>
    <Goal GoalId="2" Minute="85" OwnGoal="n">
     R.Kip</Goal>
   </GoalScorersList>
   <FinalGoals>2</FinalGoals>
  </Home>
  <Away>
   <Team>Achilles '29</Team>
   <GoalScorersList>
    <Goal GoalId="1" Minute="19" OwnGoal="n">
     B.vandeBeek</Goal>
   </GoalScorersList>
   <FinalGoals>1</FinalGoals>
  </Away>
 </Highlights>
 <Events>
  <YellowCardList>
   <YellowCard YellowCardId="1" Minute="36">
    N. Mamedov</YellowCard>
  </YellowCardList>
  <YellowRedList/>
  <OwnGoalList/>
```

Figure 1: An excerpt from the XML files generated by scraping *Goal.com*.

potentially every match; with the original version, the system could not do that, as the information about teams was contained in a limited knowledge base created by the authors of PASS.

The paper is structured as follows. Section 2 gives an overview of the original PASS system and the reports it can generate. Sections 3 and 4 describe how the templates were translated, and other issues that needed to be addressed when expanding the system to generate a new language. In Section 5 we describe how Wikidata can be a useful resource for template-based NLG systems, and how we integrated it in our version of PASS. Finally, in Section 6 we present our conclusions and possible improvements for the system.

2 The PASS NLG system

The starting point for this work was PASS (van der Lee et al., 2017), a modular, open-source and publicly-available[1] natural language generation tool. PASS is based on templates, and generates football reports in Dutch, starting from match statistics. For each match, two reports are produced, one for the supporters of each club that played the match; the aim is generating a report that uses tailored emotional language, e.g. expressing disappointment for the loss or excitement for the victory. The templates were manually created, starting from an "affective soccer corpus" (Braun et al., 2016), i.e. summaries of the matches as published by the opposing teams, in which the emotional investment is clear.

PASS reports are composed of a title, summarizing the final result, a general introduction that gives information about the opponents or the team expectations, a summary of the game course and its events, and a final debriefing with the outcome for the team.

A brief overview of the general architecture of PASS follows.

Starting data. First, the game statistics are scraped from Goal.com. Data about the match (final result and statistics for each team, who scored and when, who was given a card by the referee), the two teams, their players, the referee and the venue (such as location, name of stadium and number of attendees) is converted to XML (Figure 1).[2]

Governing module. It controls the overall process, and shuffles data across the different modules, that are run in succession.

Lookup module. It deals with the template database, providing to the following modules the templates that match the result of each team (i.e. templates for win, loss or tie).

Ruleset module. It considers the game statistics, and depending on the results decides what templates are actually usable, for example discarding those mentioning an "equalizer goal" if the score was never even during the match.

Template selection module. It selects the templates to be used among the possible templates, privileging those that are more descriptive of the match events.

Template filler module. It replaces placeholders in templates (e.g. <*stadium*>) with the corresponding piece of information in the match data.

Topic collection module. It decides what topics need to be reported for each team, and orders them chronologically according to the match data, thus ensuring that the second goal is not reported be-

[1] https://github.com/TallChris91/PASS

[2] For an exhaustive list of what is actually scraped and stored by the system, see (van der Lee et al., 2017, §3.1). The XML file of Figure 1 can be found at https://github.com/TallChris91/PASS/blob/master/InfoXMLs/AC_ACH_30102015_goal.xml

fore the first goal, for example.

Text collection module. It structures the text in the order previously mentioned, i.e. title, introduction, summary and debriefing.

Information variety module. It replaces templates that express the same information multiple times in the same report.

Reference variety module. It generates different referring expressions when repetitions are present, e.g. changing "Ajax" into "the club of manager Peter Bosz" if "Ajax" is mentioned in two subsequent sentences.

Between-text variety module. It keeps track of how the previous reports were generated, minimizing the repetition of templates across different reports.

3 Translating the templates

As mentioned in section 1, the version of PASS released by van der Lee et al. (2017) generates reports in Dutch only. To create the English version of PASS, we used Google Translate to translate the content of Dutch templates to English. After the automatic translation, each template was manually corrected, restoring garbled placeholders and fixing the mistakes that are inevitably introduced by the translation system. This allows even a person with limited knowledge of Dutch to translate the text of the templates, without requiring the intervention of a professional translator or of a bilingual person.

PASS templates consist of a string of text with one or more placeholders enclosed in angular brackets, such as "<*focus team*> verliest <*in eigen huis; home|op bezoek; away;; homeaway*> van <*final remaining players other team*> tal <*other team*>." Placeholders can be either "simple", such as <*focus team*>, where the template filler will simply insert a proper name or number, or they can contain a conditional statement (akin to a switch-case construct), such as in <*in eigen huis; home|op bezoek; away;; homeaway*>. In this case, the *homeaway* variable can contain either "home" or "away"; in the former case, this is replaced with "in eigen huis", while in the latter the text becomes "op bezoek". Some Dutch templates, their automatic translations and the revised versions are shown in Table 1. A sample report in English is shown in Figure 2.

Depending on the placeholder type and on the sentence, two problems can arise in translation:

> In Amsterdam, PSV took three points thanks to goals from Gastón Pereiro. An eager Philips Sport Vereniging won Sunday with 1-2 against Ajax.
> After 7 minutes the result was already written, thanks to Gastón Pereiro. After 10 minutes the equalizer from Amin Younes hit the net: 1-1. In the 79th minute, midfielder Gastón Pereiro decided the game by hitting the 1-2 with the help of midfielder Jorrit Hendrix.
> Referee Nijhuis was forced to give 7 yellow cards to Jeffrey Bruma, Andrés Guardado, Kenny Tete, Jürgen Locadia, Joël Veltman, Mitchell Dijks and Gastón Pereiro.

Figure 2: A sample report targeted to PSV supporters .

either the brackets are incorrectly put in the translated sentence (e.g. one of the angular brackets is missing, or spaces are introduced between the placeholder text and the brackets, confusing the template-filler module), or they are unnecessary (as in the second example of Table 1). In both cases, the correction is trivial.

Often, however, the translation has to be reworked in a more substantial way. Examples of common errors are, for example, the order of adverbs or phrases (Table 1, third example), some gender-specific pronouns, or idiomatic expressions typical of football reporting (last example in Table 1). Mistakes are inevitable with current automatic translation systems (Hofstadter, 2018), and the presence of placeholders is likely to decrease the output quality of the machine translation software, hence the relatively poor quality of the translated output. Still, manually correcting an automatic translation is more time-efficient than producing a firsthand translation (Federico et al., 2012; Green et al., 2013), and it allowed a non-native speaker of Dutch to translate the templates.

4 Language-specific code

In addition to translating the templates from Dutch to English, the generation code also needed some changes.

A new placeholder had to be introduced in the templates, to solve the problem of ordinal numbers: in Dutch, ordinals are indicated with an "e" following the number (e.g. "na de <minute>e minuut", meaning "after the <minute>th minute"). In English templates, <th> was introduced, and the code was extended to replace this placeholder with "st", "nd", "rd" or "th", depending on the preceding number.

Still in the template filler, Dutch conjunctions,

Dutch template	<*focus team*> verslaat <*other team*>: <*final home goals*>-<*final away goals*>
Translation	<*focus team*> beats <*other team*>: <*final home goals*>-<*final away goals*>
Dutch template	<*focus team*> leed afgelopen <*day*><*ochtend; morning*|*middag; afternoon*|*avond; evening;; daytime*> een <*thuis; home*|*uit; away;; homeaway*> nederlaag tegen <*other team*>.
Translation	<*focus team*> suffered last <*day*> <*morning; morning* | *afternoon; afternoon* | *evening; evening ;; daytime*> a <*home; home* | *out; away ;; homeaway*> defeat against <*other team*>.
English template	<*focus team*> suffered last <*day*> <*daytime*> <*a home; home*|*an away; away;; homeaway*> defeat against <*other team*>.
Dutch template	<*red player*> zou na <*minute*> minuten met een rode kaart het veld moeten verlaten.
Translation	<*red player*> should leave the field after <*minute*> minutes with a red card.
English template	<*red player*> had to leave the field with a red card after <*minute*> minutes.
Dutch template	<*goal scorer*> ontvangt de bal van <*assist giver*> en jaagt het leer de winkelhaak in: <*home goals*>-<*away goals*>.
Translation	<*goal scorer*> receives the ball from <*assist giver*> and chases the leather into the square: <*home goals*> - <*away goals*>.
English template	<*goal scorer*> receives the ball from <*assist giver*> and gets the ball in the back of the net: <*home goals*>-<*away goals*>

Table 1: Samples of Dutch templates, their automatic translations and the corrected English templates.

weekdays and phrases such as "geen spelers" ("no players") were translated to English. Similarly, in the reference variety module, we had to translate multiple strings in the code that deals with referring expression generation.

These language-specific code changes are fairly limited due to the similarity between the Dutch and English grammar. The whole process of porting PASS to English took about a month and a half, with most of the time spent translating the templates. Languages with a richer morphology, or a complex grammatical case system, might instead require the insertion of grammatical information in the templates, an extensive lexicon, and the development of more extensive code for sentence realization.

5 Multilingual knowledge

Since the original version of PASS was generating reports in Dutch, its focus was on the Dutch first and second leagues. Given the limited number of teams that take part in these leagues, the system relied on a small local knowledge base, manually constructed by the authors, consisting of just 37 entries. For each team it contained name, league, city of provenance, and a list of nicknames.

This solution, however, seemed inadequate for a multilingual system. Toponyms are sometimes translated (e.g. The Hague is called "Den Haag" in Dutch, while "Londen" is the Dutch name of London), and nicknames are not necessarily shared in different languages (e.g. the case of Batistuta mentioned in the Introduction). Furthermore, Goal.com is a comprehensive website, containing game results for most national leagues, so a limited hand-crafted knowledge base would cripple the potential of the system. To produce accurate reports, such a knowledge base would also need to be updated whenever a change in the leagues occurs, e.g. a team being promoted or demoted to a different league. To sidestep these issues, we decided to exploit Wikidata instead.

Wikidata[3] is a a free collaborative, multilingual database of structured data (in contrast to Wikipedia, which contains data in unstructured form). Wikidata is akin to DBpedia and similar projects. Wikidata can be seen as a collection of *items* (e.g. item `Q20110`, representing the Italian football player Francesco Totti), each consisting of a *label*, a *description* and a number of possible *aliases*. Each of these can have different language localizations; hence, while the label "Francesco Totti" will be the same for all languages,[4] the description will vary ("Italian footballer" for English, "Italiaanse voetballer" for Dutch), and so will the list of aliases (since in the Italian league nicknames are common, the list of aliases for Italian includes "il Capitano", "il Gladiatore", "er Pupone" and so on). Each item is also associ-

[3] https://www.wikidata.org

[4] That is, ignoring the transliterations for Arabic, Chinese and other alphabets and writing systems, which can also be present in Wikidata.

ated with a list of *statements*, i.e. *properties* and *values* that describe known facts about the item. Item Q20110 (Totti) has property P413 ("position played on team/speciality"), whose value is Q193592, i.e. the item "midfielder" (or "middenvelder", according to the Dutch label of entity Q193592).

The advantages of using Wikidata in a template-based system like PASS are many. It is multilingual, hence allowing the production of referring expressions such as "the club from The Hague" and "de club uit Den Haag", where every element including the city name is translated. It is a large-scale resource, containing structured knowledge that spans from major clubs, such as Ajax (Q81888), down to Cambridge United F.C. (Q18509) or NK Vinogradar (Q1348301), two teams playing in the third leagues of England and Croatia respectively. It is arguably more reliable and up-to-date than similar alternatives such as DBpedia, where content is automatically derived from Wikipedia instead of manually curated.[5] Last but not least, it seems that the football domain is among those with the most coverage (Fossati et al., 2017).

The integration of PASS and Wikidata is done in the Template Filler module; there, the MediaWiki API[6] is used to look for a string literal (e.g. "VVV"). This results in a list of items, such as Q25505492 (a student loan system), Q631778 (a magazine), Q1866807 (a women's football club from Venlo) and finally Q24689, the actual football club from the Dutch city of Venlo. To disambiguate between these entities, the algorithm looks for the first one whose property "instance of" (P31) has value "association football club" (Q476028). Once we have obtained an entity from Wikidata, information about that entity can be found in the system by looking at the relevant properties, and their label in the appropriate language -either Dutch or English- can be used to fill the template in.

For football clubs, the city of provenance, the name of the trainer and club nicknames can be found. For players, the nicknames and roles can be extracted. More information is available, and could potentially be used in future versions of the system, if the template placeholders or the referring expression generation module are extended.

[5]For an an extensive comparison between Wikidata and similar resources, see (Färber et al., 2016).

[6]WbSearchEntities and WbGetEntities in particular.

6 Conclusions and future work

The version of PASS presented in this paper is able to generate reports both in Dutch and English, and will soon be available online.

However, as described in Section 3, the current templates consist of a translation of the original Dutch templates, and hence may sometimes contain direct translations of idiomatic expressions that are not necessarily typical of English sports reporting.

In this perspective, better templates could be produced by repeating the same methodology used for creating the original PASS templates, i.e. starting from a corpus of English reports and manually annotating some sentences, replacing the entities therein contained with placeholders. The original corpus of PASS contains reports in Dutch, English and German (Braun et al., 2016), hence it would be appropriate for this task.

Moreover, as suggested by van der Lee et al. (2017), an automatic method of generating templates could be used to increase the number of templates, or to quickly introduce a new language.

Similarly, a more varied output could be obtained by adding strategies for lexical variations (Gatti et al., 2014): after generating the sentence from a template, the text is parsed and words are inserted, replaced or removed according to a language model. Hence, a template mentioning an "amazing goal" could result in a sentence describing a "great goal", or a "beautiful goal", without the need to add grammatical and semantic markers inside the templates.

In any case, an evaluation of the English generated texts should be performed, and the results compared with those of the Dutch version of PASS. As reported by van der Lee et al. (2017, 2018), readers were clearly able to distinguish the team for which a report was tailored, and found acceptable levels of clarity and fluency for the reports, while the correctness of the information given is even higher than in human-written reports. We expect the English version of PASS to obtain similar positive results.

We believe that the work here presented shows how Wikidata can be an useful resource, thanks to its extensive coverage - both in terms of knowledge and languages present - and its dynamic nature, and that it should be considered when developing multilingual NLG systems.

References

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

Nadine Braun, Martijn Goudbeek, and Emiel Krahmer. 2016. The Multilingual Affective Soccer Corpus (MASC): Compiling a biased parallel corpus on soccer reportage in English, German and Dutch. In *Proceedings of the 9th International Natural Language Generation Conference*, pages 74–78.

Stephan Busemann and Helmut Horacek. 1997. Generating air quality reports from environmental data. In *Proceedings of the DFKI Workshop on Natural Language Generation*, pages 15–21.

Karine Chevreau, José Coch, José A García-Moya, and Margarita Alonso. 1999. Generación multilingüe de boletines meteorológicos. *Procesamiento del lenguaje natural*, 25:51–58.

Ondřej Dušek and Filip Jurčíček. 2013. Robust multilingual statistical morphological generation models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Student Research Workshop*, pages 158–164.

Michael Elhadad and Jacques Robin. 1996. An overview of SURGE: A reusable comprehensive syntactic realization component. In *Proceedings of the 8th International Natural Language Generation Workshop (Posters and Demonstrations)*.

Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger. 2016. Linked data quality of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO. *Semantic Web*, 9(1):77–129.

Marcello Federico, Alessandro Cattelan, and Marco Trombetti. 2012. Measuring user productivity in machine translation enhanced computer assisted translation. In *Proceedings of the 10th Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 44–56.

Marco Fossati, Emilio Dorigatti, and Claudio Giuliano. 2017. N-ary relation extraction for simultaneous t-box and a-box knowledge base augmentation. *Semantic Web*, 9(4):413–439.

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

Lorenzo Gatti, Marco Guerini, Oliviero Stock, and Carlo Strapparava. 2014. Sentiment variations in text for persuasion technology. In *International Conference on Persuasive Technology (PERSUASIVE 2014)*.

Spence Green, Jeffrey Heer, and Christopher D. Manning. 2013. The efficacy of human post-editing for language translation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 439–448.

Douglas Hofstadter. 2018. The shallowness of Google Translate. *The Atlantic*.

Chris van der Lee, Emiel Krahmer, and Sander Wubben. 2017. PASS: A Dutch data-to-text system for soccer, targeted towards specific audiences. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 95–104.

Chris van der Lee, Bart Verduijn, Emiel Krahmer, and Sander Wubben. 2018. Evaluating the text quality, human likeness and tailoring component of PASS: A Dutch data-to-text system for soccer. In *Proceedings of the 27th International Conference on Computational Linguistics*.

Simon Mille, Miguel Ballesteros, Alicia Burga, Gerard Casamayor, and Leo Wanner. 2016. Multilingual natural language generation within abstractive summarization. In *Proceedings of the 1st International Workshop on Multimodal Media Data Analytics (in conjunction with ECAI)*, pages 33–38.

Pierre-Luc Vaudry and Guy Lapalme. 2013. Adapting SimpleNLG for bilingual English-French realisation. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 183–187.

Michael White and Rajakrishnan Rajkumar. 2009. Perceptron reranking for CCG realization. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 410–419.