WebNLG 2020 Challenge: Semantic Template Mining for Generating References from RDF

Trung Tran **Dang Tuan Nguyen**
Saigon University, Ho Chi Minh City, Vietnam
ttrung@nlke-group.net
dangnt@sgu.edu.vn

Abstract

We present in this paper our mining system for shared task WebNLG Challenge 2020. The general idea of the system is that we generate the semantic template of the output reference from the input RDF XML structure. In the training process, we perform the following subtasks: (i) extract the core information from input RDF; (ii) generate semantic templates from corresponding references. With new RDF XML data, we detect the core information, in turn add the new template into the warehouse and determine the output semantic template. We will evaluate the output natural language references in two processes: automatic and human evaluations. The results of the first tested process show that our system generates the high quality English descriptions from testing RDF XML structures and has a good contribution to the NLG state-of-the-art.

1 Introduction

Natural Language Generation (NLG) plays a critical role in the modern era. Researchers proposed different approaches to generate high-quality text from input structured data in different domains (Gatt and Krahmer, 2018; Laha et al., 2019; Moryossef et al., 2019; Shimorina and Gardent, 2018; Trisedya et al., 2018; Nguyen and Tran, 2018, 2020; Moussallem et al. 2018; Jagfeld et al., 2018; Dušek et al., 2018, 2020; Ferreira et al., 2019). Especially, with the growing need in the Semantic Web (SW) communities, there are the requirements for NLG works to provide a natural means for presenting this data in an organized, coherent and accessible way.

In the first major task of WebNLG 2020 [1] shared task, the Organizer provided English dataset for training which comprises data-text pairs for 16 distinct DBpedia[2] categories. Each entry in the dataset comprises a Resource Description Framework[3] (RDF) triple set paired with several natural language (NL) references. We illustrated an example of an English entry in the training data[1] in Table 1. The aim of this major task is to generate the appropriate NL reference for each input triple set.

RDF XML	`<entry category="Airport" size="3">` `  <modifiedtripleset>` ` <mtriple>Aarhus_Airport	location	Tirstrup</mtriple>` ` <mtriple>Tirstrup	country	Denmark</mtriple>` ` <mtriple>Denmark	language	Danish_language</mtriple>` ` </modifiedtripleset>` `</entry>`
References	• Aarhus Airport is located in Tirstrup, Denmark; where the language is Danish. • Aarhus Airport is located in Tirstrup, Denmark where the language spoken is Danish. • Aarhus Airport is located in Tirstrup, Denmark where the Danish language is spoken.

Table 1: Sample of <RDF triple set – NL references>.

The primary purpose of this article is to present our system in RDF-to-text generation task. Developing from the ideas in (Nguyen and Tran,

[1] https://webnlg-challenge.loria.fr/challenge_2020/
[2] https://wiki.dbpedia.org/
[3] https://www.w3.org/TR/rdf11-concepts/

3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+), Dublin, Ireland (Virtual), 18 December 2020, pages 177–185,

2018, 2020), we propose the mining approach that creates the intermediate semantic template from input RDF data. We represent the general architecture of our system in Fig.1 with the following main components: (i) information extraction in which we detect the core contents and classify into groups; (ii) template mining in which we analyze the above groups and input references to form the semantic template; (iii) augmentation in which we analyze the new RDF data without references to define the new template and augment to the warehouse.

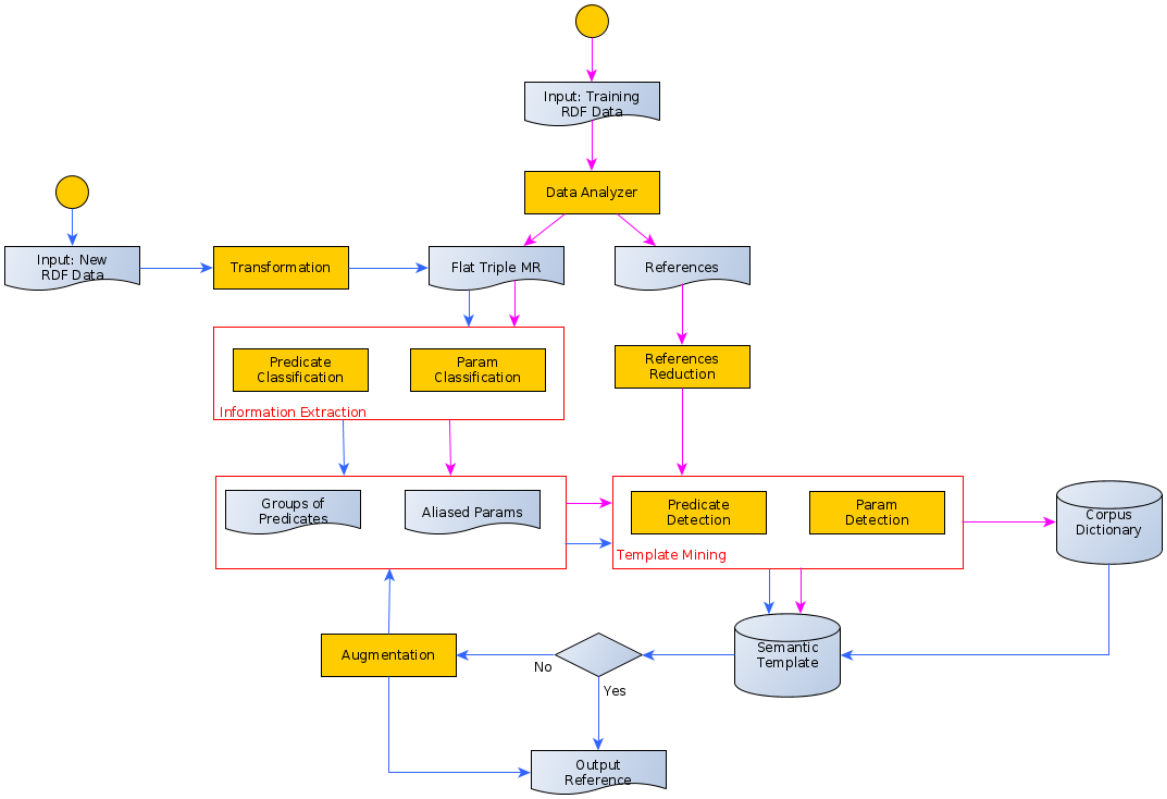

Figure 1: The general architecture of semantic template mining system.

The rest of article is separated as follows. We clarify the background knowledge in Section 2 and describe our generation system in Section 3. Section 4 details the evaluation from the Organizers and analyzes the results. We offer conclusions in Section 5.

2 Background Knowledge

The main content of this section is to clarify the background knowledge that we apply in the research. The first part is to present Flat Triple Meaning Representation (MR) which is an intermediate structure to express core information from input RDF XML data. The second part is to present Jaro-Winkler Similarity, which is used to find the phrases which have the similarity content and then we determine which one should be selected to form the templates. The third part is to present aliased parameters and semantic template which are the core information in our system.

2.1 Flat Triple Meaning Representation

Following forming Flat MR structure in the E2E Challenge 2017 (Dušek et al., 2018, 2020), we define a new structure called Flat Triple Meaning Representation (MR) which is the plaintext form of RDF XML. This form is like the representation of relationships between predicates of Flat MR structure from (Nguyen and Tran, 2018, 2020).

We define a Flat Triple MR:

- We transform each triple of RDF XML into a list of predicates of Flat Triple MR.
- Each predicate comprises: (i) the name of predicate, which is the relationship between two items; (ii) the subject parameter, which expresses the first item; (iii) the object parameter, which expresses the second item.

As an example, we can transform the RDF XML in Table 1 into Flat Triple MR:

```
location[Aarhus_Airport | Tirstrup];
country[Tirstrup | Denmark]; language[Denmark
| Danish_language];
```

In Fig.2, we illustrate the relationships between parameters and predicates in the above Flat Triple MR:

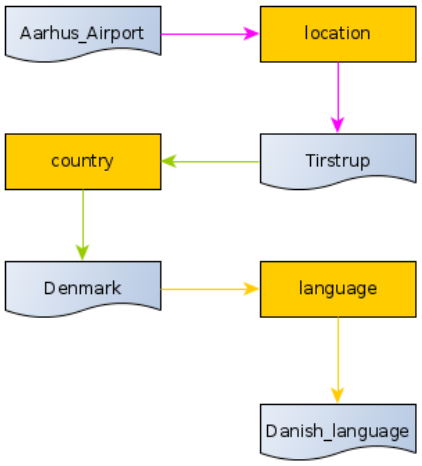

Figure 2: The relationship between each predicate and its parameters.

2.2 Jaro-Winkler Similarity

Jaro Similarity (Jaro, 1989, 1995; Winkler, 1990, 2006; Cohen et al., 2003) is the measure of similarity between two strings. We calculate the value of Jaro distance in the range [0, 1]. From this range, the value 1 means the strings are equal and the value 0 means these two strings do not have similarity at all.

The Jaro Similarity (Wikipedia, 2020) is calculated with the following formula:

$$Jaro\_sim(s_1, s_2) = \begin{cases} 0, & if\ m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right), & otherwise \end{cases}$$

In the above formula: (i) $|s_i|$ is the length of s_i; (ii) m is the number of "matching characters"; (ii) t is the number of "transpositions".

The characters of s_1 and s_2 respectively, are considered matching only if they are the same and not farther than:

$$\left\lfloor \frac{\max(|s_1|, |s_2|)}{2} \right\rfloor - 1$$

Winkler (1990) introduced the Jaro-Winkler similarity[5], which is a modification of the Jaro similarity. The author placed more weight on matching the first i characters. If l is the largest number such that the first l characters of s_1 match those of s_2, then the Jaro-Winkler similarity is defined as:

$$Sim_{JW}(s_1, s_2) = Sim_J(s_1, s_2) + lp[1 - Sim_J(s_1, s_2)]$$

In the above formula, p is a constant scaling factor for how much the score is adjusted upwards for having common prefixes.

We apply the Jaro-Winkler similarity at the following actions:

- Determine the rate of similarity between corresponding references of each Flat Triple MR. We then keep only the most frequent references.
- Determine the groups of words that express each predicate and corresponding parameters.
- Determine the phrases that have the similar meaning when handling new Flat Triple MR.

2.3 Semantic Template and Aliased Parameters

Developing from ideas in (Nguyen and Tran, 2018, 2020; Gardent et al. 2017; Ferreira et al. 2019) about generating intermediate templates from input structured-data, we define special semantic templates for this research. These templates take the role is the intermediate structures of the final references.

To reduce the dependence on vocabulary, especially when handling cross domains, we define new aliases for parameters in each Flat Triple MR. The idea to determine new aliases is described as follows:

- Analyzing each Flat Triple MR paired with corresponding NL references, we found that there is one item taking the central role. Other items have relationships with this one and with each other in some levels.
- The item taking the central role will have the alias AGENT.
- The other items will have the alias PATIEN_X, with X = [1, n], n < total number of items.

As an example, the Flat Triple MR in Fig.2 can be transformed into a new one as follows:

```
location[AGENT | PATIENT_1]; country[PATIENT_1
| PATIENT_2]; language[PATIENT_2 | PATIENT_3];
```

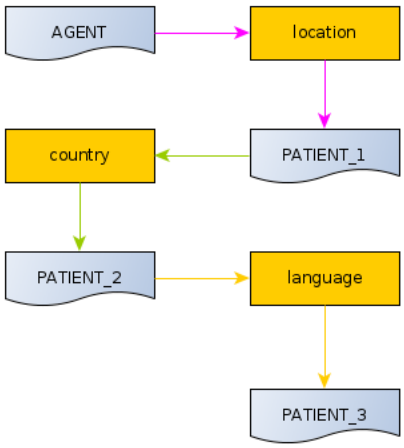

Figure 3: The relationship between each predicate and its aliased parameters.

In our system, the semantic templates are the core information, which represents the structured form of the output NL references. Each semantic template comprises three principal components:

- Aliased parameters from Flat Triple MR.
- Groups of words that express predicates from Flat Triple MR.
- Linking words that connect phrases in the output text.

As an example, the references in Table 1 have the following semantic templates with aliased parameters from Table 3:

- **AGENT** is located in **PATIENT_1**, **PATIENT_2**; where the language is **PATIENT_3**.
- **AGENT** is located in **PATIENT_1**, **PATIENT_2** where the language spoken is **PATIENT_3**.
- **AGENT** is located in **PATIENT_1**, **PATIENT_2** where the **PATIENT_3** is spoken.

3 Semantic Template Mining System

We present in this section the principal components of our system. As illustrated in the general architecture in Fig. 1, there are two processes to perform:

- In the training process, we (i) extract the information from Flat Triple MR, and (ii) build the warehouse that contains the semantic templates from the mining component.
- When analyzing new RDF data, we (i) extract the information from Flat Triple MR, (ii) determine the semantic template, or (iii) create the new semantic template and augment to the warehouse.

3.1 Information Extraction Component

The first phase for realizing this step is to determine which item in all predicates should take the alias AGENT. We handle this phase through following steps:

- **Step 1**. Determine the frequency of each item in all predicates from left to right.
- **Step 2**. Determine the items that have the highest frequency.
- **Step 3**. Determine the first item taking the alias AGENT which satisfies: (i) has the highest frequency; (ii) has the highest number of times taking subject parameter position.

To determine the alias PATIEN_X for other items, we perform two steps:

- **Step 1**. Consider each item from left to right that is not AGENT.
- **Step 2**. Set the alias PATIEN_X for this item and increase X.

As an example, the Flat Triple MR with aliased parameters illustrated in Fig. 3 results from applying the above method for the original Flat Triple MR in Fig. 2.

In the second phase of this component, we classify predicates into distinct groups. The reason for performing this phase is to better understand the grammatical structures of references. Therefore, when analyzing new RDF data in unknown domains, we could define the appropriate new semantic templates.

In this study, we define four groups of predicates. With each group, we clarify the general English grammatical structure for all predicates.

- **Group 1**. The predicates in this group show the situations in which the AGENT takes the object role of the action performed by the PATIENT_X.

As an example, we have Flat Triple MR "operatingOrganisation[**AGENT** | **PATIENT_1**]". The predicate "operatingOrganisation" indicates that **AGENT** takes the object role and **PATIENT_1** takes the subject role of the action. One semantic template for this Flat Triple MR is "**AGENT** is operated by **PATIENT_1**.".

- **Group 2**. The predicates in this group show the situations in which the PATIENT_X is the property / location / career of the AGENT.

As an example, we have Flat Triple MR "location[**AGENT** | **PATIENT_1**]". The predicate

"location" indicates that **PATIENT_1** is the location of **AGENT**. One semantic template for this Flat Triple MR is "**AGENT** is located in **PATIENT_1**.".

- **Group 3**. The predicates in this group show the situations in which the PATIENT_X is the date time.

As an example, we have Flat Triple MR "birthDate[**AGENT** | **PATIENT_1**]". The predicate "birthDate" indicates that **PATIENT_1** is the date time. One semantic template for this Flat Triple MR is "**AGENT** was born on **PATIENT_1**.".

- **Group 4.** The remaining predicates should be in this group.

As an example, we have Flat Triple MR "almaMater[**AGENT** | **PATIENT_1**]". One semantic template for this Flat Triple MR is "**AGENT** was graduated from the **PATIENT_1**.".

3.2 Template Mining Component

We build the warehouse of semantic templates and dictionary of corpora in this component. The crucial question here is: How to determine the strings (groups of words) that express the information of each parameter or predicate? To answer this question, we perform the following sub-tasks:

During the first sub-task, we analyze the parameter with the following steps:

- **Step 1**. Split the parameter into a set of separated tokens.
- **Step 2**. Adding the prepositions to the suitable position in the above set. We handle this step according to training references and common English communications. We then create the new string from the set of token in each situation. We will add the new string into the dictionary.
- **Step 3**. We determine the number of tokens in each string. We then sort the list of strings according to the number of tokens from highest to lowest.

As an example, with parameter "Jones_County,_Texas", we can have strings: "Jones County Texas" or "Jones County, Texas".

During the second sub-task, we determine the strings in reference that have the similarity with each string in the list from the above first task, which expresses the current considering parameter. We perform this task through the following steps:

- **Step 1**. We build the list of n-grams from the considering reference. Here, n is the number of tokens of each string from the list in the above first sub-task. We then create the string for each n-gram.
- **Step 2.** We browse each string from Step 1 and consider two situations:
  - **Step 2.1**. If the current string is normal type. We apply the Jaro-Winkler similarity[5] to compare with the considering string from the list in the above first sub-task. If the similarity is higher than the threshold is 0.9, then we (i) add into the dictionary and (ii) replace by the alias of current considering parameter.
  - **Step 2.2**. If the current string is date type. We transform this string and the considering string from the list in the above first sub-task into date format. We then check if these two dates are the same or not. If they are the same dates, then we (i) add into the dictionary and (ii) replace by the alias of current considering parameter.

As an example, with the above defined strings, we determine some strings from training references: "Jones County in Texas". Another example is parameter "1913-05-05", we determine the string that expressed the same date is "May 5th 1913" or "May 5, 1913".

During the third sub-task, we detect the groups of words that express each predicate with the same idea as in the above first and second sub-task.

- **Step 1**. We analyze the current predicate. We create a string that contains all tokens extracted from this predicate.
- **Step 2**. We build three lists of n-grams from the considering reference. Here n is in turn one of three numbers: (i) number of the above tokens – 1; (ii) number of the above tokens; (iii) number of the above tokens + 1. We then create the string for each n-gram.
- **Step 3**. We apply the Jaro-Winkler similarity[5] to compare with the string from Step 1. If the similarity is higher than the threshold is 0.8, then we (i) add into the dictionary and (ii) replace by the alias of current considering parameter.

As an example, with predicate "cityServed", we have some similar strings: "city is served by" or "serves the city".

3.3 Augmentation Component

We realize this step through two main phases:

- **Phase 1**. We assign alias AGENT and PATIENT_X for each parameter and classify predicates into groups.

As an example, consider Flat Triple MR "producer[English_Without_Tears | Anatole_de_Grunwald]" in the surprise domain. We have the alias Flat Triple MR "producer[**AGENT** | **PATIENT_1**]". The predicate "producer" is classified into Group 2, as mentioned in Section 3.1.

- **Phase 2**. Based on the information from Phase 1, we define the new semantic templates for input Flat Triple MR having 1 to 7 predicates.

As an example, with the above Flat Triple MR, we the find the new semantic templates: (i) "The producer of **AGENT** is **PATIENT_1**."; (ii) "**AGENT** was produced by **PATIENT_1**.".

The key ideas for handling phase 2 are:

- As mentioned in Section 3.1, with each group of predicate, we have the similar English grammatical structures.
- We see that the complex semantic templates in warehouse, which are the output of Flat Triple MR having more than one predicate, are actually different grammatical structures to express the combination between other semantic templates, which are the output of Flat Triple MR having less number of predicates. We define the new complex structures based on this observation and common English communication.

4 Experiment and Evaluation

The Organizer provided the testing data with three types of characteristics: (i) all the entities and categories of RDF triples/texts existed in the training data; (ii) only categories of RDF triples/texts existed in the training data and not the entities; (iii) surprise domains in which the categories do not exist in the training data. The total amount of testing data can be classified into categories as in Table 2:

Categories	Surprise Domain	Number Entries	Percentage on Total Entries
Airport		95	5.34%
Artist		109	6.13%
Astronaut		82	4.61%
Athlete		50	2.81%
Building		46	2.59%
Celestial Body		49	2.75%
City		83	4.65%
Comics Character		30	1.69%
Company		66	3.71%
Film	X	264	14.84%
Food		46	2.59%
Mean Of Transportation		58	3.26%
Monument		46	2.59%
Musical Work	X	290	16.30%
Politician		29	1.63%
Scientist	X	259	14.56%
Sports Team		44	2.47%
University		90	5.06%
Written Work		43	2.42%

Table 2: Brief analysis of testing data.

4.1 Automatic Evaluation

According to the WebNLG 2020 Challenge, to measure the scores, the Organizer used five main metrics: BLEU (Papineni et al. 2002), METEOR (Lavie and Agarwal 2007), chrF++ (Popović, 2015, 2017), TER (Snover et al. 2006), and BERT-Score (Zhang et al. 2020). There are total 33 guess systems, including 2 baseline systems for the comparison. According to the results, our system ranks 9th when ordered by METEOR metric, which means our system gets better METEOR than 24 other submissions, including 2 baseline systems. Besides, when comparing with 2 baseline systems, our system gets better points in most of the metrics.

In Table 3, we show the results of the top ten systems according to the automatic evaluation results[4] and two baseline systems (take the rank 15 and 18 respectively).

SYSTEM ID	BLEU	BLEU NLTK	METEOR	CHRF++	TER	BERT PRECISION	BERT RECALL	BERT F1	BLEURT
id18	53.98	0.535	0.417	0.690	0.406	0.960	0.957	0.958	0.62
id30	53.54	0.532	0.414	0.688	0.416	0.958	0.955	0.956	0.61
id30_1	52.07	0.518	0.413	0.685	0.444	0.955	0.954	0.954	0.58
id34*	52.67	0.523	0.413	0.686	0.423	0.957	0.955	0.956	0.6
id5	51.74	0.517	0.411	0.679	0.435	0.955	0.954	0.954	0.6
id35*	51.59	0.512	0.409	0.681	0.431	0.956	0.954	0.954	0.59

id23	51.74	0.514	0.403	0.669	0.417	0.959	0.954	0.956	0.61
id2	50.34	0.500	0.398	0.666	0.435	0.954	0.950	0.951	0.57
id15	***40.73***	***0.405***	***0.393***	***0.646***	***0.511***	***0.940***	***0.946***	***0.943***	***0.45***
id28	44.56	0.432	0.387	0.637	0.479	0.949	0.949	0.948	0.54
….									
Baseline	40.57	0.396	0.373	0.621	0.517	0.946	0.941	0.943	0.47
….									
baseline_2	37.89	0.371	0.364	0.606	0.553	0.933	0.935	0.930	0.42

Table 3: Results of automatic evaluation: top ten systems and two baseline systems.

The full results of automatic evaluation are showed at the final WebNLG 2020 website[4].

4.2 Human Evaluation

For human evaluation, the Organizer assesses the system outputs according to the following criteria by native speakers recruited on crowdsourcing platforms: (i) ***Data Coverage*** – how much information from the data has been covered. The text will be evaluated if it fully covers all predicates shown in the data; (ii) ***Relevance*** – this criterion evaluates if the text contains any non-presented predicates. The text will be evaluated if it mentions/describes only predicates which are in the input; (iii) ***Correctness*** – the text will be evaluated if it describes predicates (which are both in data and text, e.g. relevant predicates) with correct objects. Also, the subject has to be described correctly; (iv) ***Text Structure*** – the text will be evaluated if it is grammatical and well-structured, which means the structural/grammatical quality, written in good English; (v) ***Fluency*** – the text will be evaluated if it progresses naturally and sounds like a coherent whole, which means the "naturalness".

Each criterion has been rated with a single number in the range from "0" (completely disagree) to "100" (completely agree). The scores as they appear for each criterion have been normalised (z-scores) and clustered into groups among which there are no statistically significant differences according to the Wilcoxon rank-sum significant test.

According to the results of human evaluation, there are total 17 systems, including 2 baseline systems, which were evaluated in this phase. Our system ranks 1st in the first three criteria (Data Coverage, Relevance, Correctness), ranks 3rd in Text Structure criterion and ranks 4th in Fluency criterion. In Table 4, we show the results of our system (DANGNT-SGU) compared with two baseline systems as well as five other systems which rank 1st in at least three criteria, according to human evaluation phase. The results are ordered alphabetically. The full results can be viewed at the final WebNLG 2020 website[4].

As can be seen in Table 4, our system gets better scores in most of the criteria than two baseline systems, especially in the first three criteria (Data Coverage, Relevance, Correctness). Besides, there are two systems, which are AmazonAI and OSU_Neural_NLG, rank 1st in all criteria.

The testing results show that our system generates good quality references from RDF structures in WebNLG 2020 Challenge experiment sections. Based on cursory checks, our system was able to create long, grammatical, meaningful, multi-sentence output, as illustrated by the following example:

Flat Triple MR	populationDensity[Ciudad_Ayala | 1604.0]; leaderTitle[Ciudad_Ayala | Governator]; country[Ciudad_Ayala | Mexico]; elevationAboveTheSeaLevel[Ciudad_Ayala | 1147.0]; timeZone[Ciudad_Ayala | Pacific_Daylight_Time];
Reference	Ciudad Ayala, which is in the time zone of Pacific Daylight Time, is led by the Governator in Mexico. It has a population density of 1604.0 and an elevation of 1147.0 above sea level.

[4] https://beng.dice-research.org/gerbil/

System	Data Coverage		Relevance		Correctness		Text Structure		Fluency	
	Rank	Avg. Z	Rank	Avg. Z	Rank	Avg. Z	Rank	Avg. Z	Rank	Avg. Z
AmazonAI	1	0.207	1	0.193	1	0.216	1	0.24	1	0.31
BASELINE	**2**	**0.143**	**2**	**0.097**	**2**	**0.131**	**2**	**0.047**	**3**	**0.022**
BASELINE2017	**2**	**0.096**	**1**	**0.193**	**3**	**0.079**	**2**	**0.028**	**3**	**-0.082**
bt5	2	0.06	1	0.134	1	0.146	1	0.177	2	0.13
DANGNT-SGU	***1***	***0.228***	***1***	***0.153***	***1***	***0.136***	***3***	***-0.153***	***4***	***-0.162***
NUIG-DSI	2	0.082	1	0.113	1	0.18	1	0.221	1	0.2
OSU_Neural_NLG	1	0.215	1	0.11	1	0.18	1	0.238	1	0.229
RALI	1	0.268	1	0.153	1	0.179	3	-0.211	4	-0.156
REF	2	0.173	1	0.112	1	0.181	1	0.162	2	0.181

Table 4: Results of human evaluation: our system and two baseline systems.

5 Conclusion

We have presented our mining system for generating English natural language references from XML RDF structure. Our approach has three most important sub-tasks: (i) extract the core information from input RDF; (ii) detect the group for each relationship and aliases for its parameters and generate semantic template; (iii) determine new information and augment to the warehouse. The evaluation results show that our approach overcomes the requirements: (i) references have lexical richness, syntactic variation, and discourse phenomena; (ii) references cover whole contents from the input RDF structure.

In future works, we intend to apply techniques in Deep Learning and knowledge in linguistic theories to improve the quality and naturalness of generated texts. Besides, we expand our approach to other datasets for a broader comparison.

References

Emilie Colin, Claire Gardent, Yassine M'rabet, Shashi Narayan and Laura Perez-Beltrachini. 2016. The WebNLG Challenge: Generating Text from DBPedia Data. In *Proceedings of INLG*. Edinburgh (UK).

William W. Cohen, Pradeep Ravikumar and Stephen E. Fienberg. 2003. A comparison of string distance metrics for name-matching tasks. *KDD Workshop on Data Cleaning and Object Consolidation*, 3: 73–78.

Ondřej Dušek, Jekaterina Novikova and Verena Rieser. 2018. Findings of the E2E NLG Challenge. In *Proceedings of INLG*. Tilburg (The Netherlands).

Ondřej Dušek, Jekaterina Novikova and Verena Rieser. 2020. Evaluating the state-of-the-art of End-to-End Natural Language Generation: The E2E NLG challenge. *Computer Speech & Language*, 59:123-156.

Thiago Castro Ferreira, Diego Moussallem, Emiel Krahmer and Sander Wubben. 2018. Enriching the WebNLG corpus. In *Proceedings of INLG*. Tilburg (The Netherlands).

Thiago Castro Ferreira, Chris van der Lee, Emiel van Miltenburg and Emiel Krahmer. 2019. Neural data-to-text generation: A comparison between pipeline and end-to-end architectures. In *Proceedings of EMNLP*. Hong Kong (China).

Thiago Castro Ferreira, Claire Gardent, Nikolai Ilinykh, Chris van der Lee, Simon Mille, Diego Moussalem and Anastasia Shimorina. 2020. The 2020 Bilingual, Bi-Directional WebNLG+ Shared Task: Overview and Evaluation Results (WebNLG+ 2020). In *Proceedings of the 3rd WebNLG Workshop on Natural Language Generation from the Semantic Web (WebNLG+ 2020)*.

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*, 61:737–763.

Claire Gardent, Anastasia Shimorina, S. Narayan and Laura Perez-Beltrachini. 2017. Creating Training Corpora for NLG Micro-Planners. In *Proceedings of ACL*. Vancouver (Canada).

Claire Gardent, Anastasia Shimorina, Shashi Narayan and Laura Perez-Beltrachini. 2017. The WebNLG challenge: Generating text from RDF data. In *Proceedings of INLG*. Santiago de Compostela (Spain).

Glorianna Jagfeld, Sabrina Jenne and Ngoc Thang Vu. 2018. Sequence-to-Sequence Models for Data-to-Text Natural Language Generation: Word- vs. Character-based Processing and Output Diversity. In *Proceedings of INLG*. Tilburg (The Netherlands).

Matthew A. Jaro. 1989. Advances in record linkage methodology as applied to the 1985 census of Tampa Florida. Journal of the *American Statistical Association*. 84(406): 414–420.

Matthew A. Jaro. 1995. Probabilistic linkage of large public health data file. *Statistics in Medicine*. 14(5–7): 491–498.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*. Prague (Czech Republic).

Anirban Laha, Parag Jain, Abhijit Mishra and Karthik Sankaranarayanan. 2019. Scalable Micro-planned Generation of Discourse from Structured Data. Journal of *Computational Linguistics*, 45(4):65–170.

Amit Moryossef, Yoav Goldberg and Ido Dagan. 2019. Step-by-Step: Separating Planning from Realization in Neural Data-to-Text Generation. In *Proceedings of NAACL-HLT*. Minneapolis, Minnesota (USA).

Diego Moussallem, Thiago Castro Ferreira, Marcos Zampieri, Maria Claudia Cavalcanti, Geraldo Xexeo, Mariana Neves and Axel-Cyrille Ngonga Ngomo. 2018. RDF2PT: Generating Brazilian Portuguese Texts from RDF Data. In *Proceedings of LREC*. Miyazaki (Japan).

Diego Moussalem, Paramjot Kaur, Thiago Castro Ferreira, Chris van der Lee, Anastasia Shimorina, Conrads, Felix, Michael Röder, René Speck, Claire Gardent, Simon Mille, Nikolai Ilinykh and Axel-Cyrille Ngonga Ngomo. 2020. A General Benchmarking Framework for Text Generation. In *Proceedings of the 3rd WebNLG Workshop on Natural Language Generation from the Semantic Web (WebNLG+ 2020)*.

Dang Tuan Nguyen and Trung Tran. 2018. Structure-based Generation System for E2E NLG Challenge. In *E2E NLG Challenge System Descriptions*.

Dang Tuan Nguyen and Trung Tran. 2020. A Template-based Approach for Generating Vietnamese References from Flat MR Dataset in Restaurant Domain. In *Proceedings of FDSE*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia (USA).

Laura Perez-Beltrachini, Rania Mohamed Sayed and Claire Gardent. 2016. Building RDF Content for Data-to-Text Generation. In *Proceedings of COLING*. Osaka (Japan).

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*. Lisbon (Portugal).

Maja Popović. 2017. chrF++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, Volume 2: Shared Tasks Papers. Copenhagen (Denmark).

Ehud Reiter and Robert Dale. 2000. Building Natural Language Generation Systems, studies in natural language processing edition. Cambridge University Press.

Anastasia Shimorina, Elena Khasanova and Claire Gardent. 2019. Creating a Corpus for Russian Data-to-Text Generation Using Neural Machine Translation and Post-Editing. In *Proceedings of BSNLP Workshop*. Florence (Italy).

Anastasia Shimorina and Claire Gardent. 2018. Handling Rare Items in Data-to-Text Generation. In *Proceedings of INLG*. Tilburg (The Netherlands).

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas*.

Bayu Distiawan Trisedya, Jianzhong Qi, Rui Zhang and Wei Wang. 2018. GTR-LSTM: A Triple Encoder for Sentence Generation from RDF Data. In *Proceedings of ACL*. Melbourne (Australia).

William E. Winkler. 1990. String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. In *Proceedings of the Section on Survey Research Methods.* American Statistical Association: 354–359.

William E. Winkler. 2006. Overview of Record Linkage and Current Research Directions. Research Report Series, RRS.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. In *Proceedings ICLR*.

Wikipedia contributors. 2020. Jaro-Winkler distance. *Wikipedia, The Free Encyclopedia*.