Generating Paraphrases from DBPedia using Deep Learning

Amin Sleimi
Université de Lorraine, Nancy (France)
amin.sleimi@gmail.com

Claire Gardent
CNRS/LORIA, Nancy (France)
claire.gardent@loria.fr

Abstract

Recent deep learning approaches to Natural Language Generation mostly rely on sequence-to-sequence models. In these approaches, the input is treated as a sequence whereas in most cases, input to generation usually is either a tree or a graph. In this paper, we describe an experiment showing how enriching a sequential input with structural information improves results and help support the generation of paraphrases.

1 Introduction

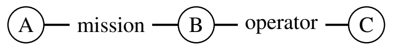
Following work by (Karpathy and Fei-Fei, 2015; Kiros et al., 2014; Vinyals et al., 2015; Fang et al., 2015; Xu et al., 2015; Devlin et al., 2014; Sutskever et al., 2011; Bahdanau et al., 2014; Luong et al., 2014), there has been much work recently on using deep learning techniques to generate text from data. (Wen et al., 2015) uses recurrent neural network to generate text from dialog speech acts. Using biography articles and infoboxes from the WikiProject Biography, (Lebret et al., 2016) learns a conditional neural language model to generate text from infoboxes. etc.

A basic feature of these approaches is that both the input and the output data is represented as a *sequence* so that generation can then be modeled using a Long Short Term Memory Model (LSTM) or a conditional language model.

Mostly however, the data taken as input by natural language generation systems is *tree or graph structured*, not linear.

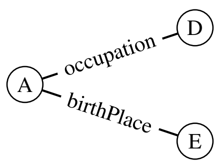
In this paper, we investigate a constrained generation approach where the input is enriched with constraints on the syntactic shape of the sentence to be generated. As illustrated in Figure 1, there is a strong correlation between the shape

T_1



$S_{1.1}$ *A participated in mission B operated by C*

$S_{1.2}$ *A participated in mission B which was operated by C*



T_2

$S_{2.1}$ *A was born in E. She worked as an engineer.*

$S_{2.2}$ *A was born in E and worked as an engineer.*

Figure 1: Input and Output Shapes (A = Susan Helms, B = STS 78, C = NASA, D = engineer, E = Charlotte, North Carolina).

of the input and the shape of the corresponding sentence. The chaining structure T_1 where B is shared by two predications (mission and operator) will favour the use of a participial or a passive subject relative clause. In contrast, the tree structure T_2 will favour the use of a new clause with pronominal subject or a coordinated VP. Using synthetic data, we explore different ways of integrating structural constraints in the training data. We focus on the following two questions.

1. Does structural information improve performance ?

We compare an approach where the structure of the input and of the corresponding paraphrase is made explicit in the training data with one where it is left implicit. We show that a model trained on a corpus making this information explicit helps improve the quality of the generated sentences.

2. Can structural information be used to generate paraphrases ?

Our experiments indicates that training on corpora making explicit structural information in the input data permits generating not one but several sentences from the same input.

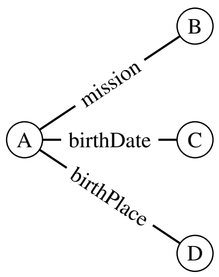


Figure 2: Example Input Graph (Subject and Object names have been replaced by capital letters)

In this first case study, we restrict ourselves to input data of the form illustrated in Figure 2 (i.e., input data consisting of three DBPedia triples related by a shared subject $(e\ p1\ e1)$ $(e\ p2\ e2)$ $(e\ p3\ e3)$) and explore different strategies for learning to generate paraphrases using the sequence-to-sequence model described in (Sutskever et al., 2011).

2 Training Corpus

To learn our sequence-to-sequence models for generation and to test our hypotheses, we build a synthetic training data-to-text corpus for generation which consists of 18 397 (data,text) pairs split into 11039 pairs for training, 7358 for development and 7358 for testing.

We build this corpus by extracting data from DBPEdia using SPARQL queries and by generating text using an existing surface realiser. As a result, each training item associates a given input shape (the shape of the RDF tree from DBPedia) with several output shapes (the syntactic shapes of the sentences generated from the RDF data by our surface realiser). Figure 3 shows an example input data and the corresponding paraphrases.

2.1 Data

RDF triples consist of (subject property object) tuples such as (Alan Bean occupation Test pilot). As illustrated in Figure 1, RDF data can be represented by a graph in which edges are labelled with properties and vertices with subject and object resources.

To construct a corpus of RDF data units which can serve as input for NLG, we retrieve sets of RDF triples from DBPedia SPARQL endpoint. Given a DBPedia category (e.g., Astronaut), we define a SPARQL query that searches for all entities of this category which have a given set of properties. The query then returns all sets of RDF triples which satisfy this query. For instance, for the category Astronaut , we use the SPARQL query shown in Figure 4. Using this query, we extract sets of DBPedia triples corresponding to 634 entities (astronauts).

2.2 Text

To associate data with text, we build lexical entries for DBPedia properties and use a small handwritten grammar to automatically generate text from sets of DBPedia triples using the GenI generator (Gardent and Kow, 2007).

Lexicon. The lexicon is constructed semi-automatically by tokenizing the RDF triples and creating a lexical entry for each RDF resource. Subject and Object RDF resources trigger the automatic creation of a noun phrase where the string is simply the name of the corresponding resource (e.g., *John E Blaha, San Antonio, ...*). For properties, we manually create verb entries and assign each property a given lexicalisation. For instance, the property birthDate is mapped to the lexicalisation *was born on*.

Grammar. We use a simple Feature-Based Lexicalised Tree Adjoining Grammar which captures canonical clauses (1a), subject relative clauses (1b), VP coordination (1c) and sentence coordination (1d). Given this grammar, the lexicon described in the previous section and the RDF triple shown in (1a), the GenI generator generates the five verbalisations shown in five (1b-f).

(1) a. *John E Blaha was born on 1942 08 26*

b. *John E Blaha who was born in San Antonio worked as a fighter pilot*

c. *John E Blaha was born on 1942 08 26 and worked as a fighter pilot.*

d. *John E Blaha was born on 1942 08 26. He is from United States*

e. *John E Blaha was born on 1942 08 26 . He was born in San Antonio and worked as a fighter pilot*

Input	(JohnBlaha birthDate 1942_08_26) (JohnBlaha birthPlace SanAntonio) (JohnBlaha occupation Fighterpilot)
Simpl.Input	JohnBlaha birthDate 1942_08_26 birthPlace SanAntonio occupation Fighterpilot
S1	*John Blaha who was born on 1942 08 26 was born in San Antonio. He worked as Fighter pilot*
S2	*John Blaha was born on 1942 08 26 and worked as Fighter pilot. He was born in San Antonio*
S3	*John Blaha was born on 1942 08 26 and was born in San Antonio. He is from United States*
S4	*John Blaha was born on 1942 08 26. He was born in San Antonio and worked as Fighter pilot*
S5	*John Blaha was born on 1942 08 26 . He is from United States and was born in San Antonio*
C-Input	JohnBlaha (birthDate 1942_08_26) birthPlace SanAntonio . occupation Fighterpilot

Figure 3: Example Data, Associated Paraphrases and Constrained Input from the Training Corpus

```
[
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?x ?birthDate (SAMPLE(?bP) as ?birthPlace)
?deathDate (SAMPLE(?dP) as ?deathPlace) ?occupation
?status ?nationality ?mission
                        WHERE {

            ?x rdf:type <http://dbpedia.org/ontology/Astronaut> .
            OPTIONAL {?x dbpedia2:birthPlace ?bP . }
            OPTIONAL {?x dbpedia2:birthDate ?birthDate .}
            OPTIONAL {?x dbpedia2:deathPlace ?dP .}
            OPTIONAL {?x dbpedia2:deathDate ?deathDate .}
    OPTIONAL {?x dbpedia2:occupation ?occupation .}
    OPTIONAL {?x dbpedia2:status ?status .}
    OPTIONAL {?x dbpedia2:nationality ?nationality .}
    OPTIONAL {?x dbpedia2:mission ?mission .}

            }
]
```

Figure 4: The sparql query to DBPedia endpoint for the Astronaut corpus

3 Learning

To learn a sequence-to-sequence model that can generate sentences from RDF data, we use the neural model described in (Sutskever et al., 2011) and the code distributed by Google Inc[1].

We experiment with different versions of the training corpus.

Raw corpus (BL). This is our a baseline system. In this case, the model is trained on the corpus of (data,text) pairs as is. No explicit information about the structure of the output is added to the data.

Raw Corpus+Structure Identifier (R+I). Each input data is associated with a structure identifier corresponding to one of the five syntactic shapes shown in Figure 3.

Raw corpus+Infix Connectors (R+C). The input data is enriched with infix connectors where & specifies conjunction, parentheses indicate a relative clause and "." sentence segmentation. The last line in Figure 3 shows the R+C input for S1.

4 Evaluation and Results.

We evaluate the results by computing the BLEU-4 score of the generated sentences against the reference sentence. Table 1 shows the results.

The baseline and the R+I model have very low results. For the baseline model, this indicates that training on a corpus where the same input is associated with several distinct paraphrases make it difficult to learn a good data-to-text generation model.

The marked difference between the R+I and the RI+C model shows that simply associating each input with an identifier labelling the syntactic structure of the associated sentence is not sufficient to learn a model that should predict different syntactic structures for differently labelled inputs. Interestingly, training on a corpus where the input data is enriched with infixed connectors giving indications about the structure of the associated sentence yields much better results.

5 Conclusion

Using synthetic data, we presented an experiment which suggests that enriching the data input to generation with information about the corresponding sentence structure (i) helps improve performance and (ii) permits generating paraphrases.

Further work involves threee main directions.

First, the results obtained in this first case study should be tested for genericity . That is the synthetic data approach we presented here should be tested on a larger scale taking into account input structures of different types (chaining vs branching) and different sizes.

Second, the approach should be extended and tested on "real data" i.e., on a training corpus where the DBPEdia triples used as input data are associated with sentences produced by humans and where there is consequently, no direct information about their structure.

Third, we plan to investigate how various deep learning techniques, in particular, recursive neural networks, could be used to capture the correlation between input data and sentence structure.

[1] `https://github.com/tensorflow/tensorflow/tree/master/tensorflow/models/rnn/translate`

System	S1	S2	S3	S4	S5
BL	3.6	5.9	6.6	5.9	7.5
R+I	4.0	6.5	6.9	6.5	8.2
R+C	98.2	91.7	91.6	88.8	89.1

Table 1: BLEU-4 scores

Acknowledgments

We thank the French National Research Agency for funding the research presented in this paper in the context of the WebNLG project ANR-14-CE24-0033.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473.*

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard M Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *ACL (1)*, pages 1370–1380. Citeseer.

Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. 2015. From captions to visual concepts and back. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1473–1482.

C. Gardent and E. Kow. 2007. A symbolic approach to near-deterministic surface realisation using tree adjoining grammar. In *ACL07*.

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3128–3137.

Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. 2014. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*.

R. Lebret, D. Grangier, and M. Auli. 2016. Generating Text from Structured Data with Application to the Biography Domain. *ArXiv e-prints*, March.

Minh-Thang Luong, Ilya Sutskever, Quoc V Le, Oriol Vinyals, and Wojciech Zaremba. 2014. Addressing the rare word problem in neural machine translation. *arXiv preprint arXiv:1410.8206*.

Ilya Sutskever, James Martens, and Geoffrey E Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned ltsm-base natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Computational Linguistics*, pages 1711–1721.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*.