A sequence to sequence transformer data logic experiment

Danxin Cui *
danxin.cui
@sorbonne-nouvelle.fr

Dominique Mariko *
dmariko
@yseop.com

* equal contribution

Estelle Labidurie
elabidurie
@yseop.com

Hugues de Mazancourt
hdemazancourt
@yseop.com

Patrick Paroubek
pap@limsi.fr

Abstract

In this paper we present experiments to evaluate how a T5 model behaves with regard to input data fidelity. The rationale behind these experiments is to evaluate if a sequence to sequence transformer can be constrained into generating the specifics of a financial report, and more generally whether it can trustfully reproduce a semantic logic, and to what extent.

1 Introduction

T5 by Raffel et al. (2020) recently demonstrated strong constrained and data to text generation capabilities. Experiments have been lead on AQG tasks (Grover et al. (2021)) and on the WebNLG dataset as to explore the data to text capabilities of a T5 model. In particular, Kale and Rastogi (2020) demonstrates T5 model shows interesting capacities in generalization to new domains and relations, and Kasner and Dusek (2020) proposes significant text generation experiment even without any in-domain examples.

Text generation in Finance can be very demanding as to the level of constraint a neural model should comply with. The objective of our experiment is to evaluate which data formalism we should build to achieve similar results as the results achieved by T5 models on WebNLG tasks.

In order to produce this evaluation, we chose to create a data set focusing on semantic intentions. Intentions are objects describing Natural Language Generation (NLG) pipelines based on Abstract Categorical Grammars semantic and syntactic items, as defined by Salmon (2017), that can be specialized and combined together. We also implemented a set of metrics for NLG evaluation based on BLEU and BERT-SCORE.

2 Corpus

The initial corpus for this experiment is a set of 4159 public online US and UK Market Reports. We limit the experiments to the financial domain as to prove more accurate results.

2.1 Corpus generation

Raw text is extracted with a home made pdf extractor based on PDFMiner, deleting all tables and titles. A corpus analysis is performed on these raw extractions, leading to the definition of handcrafted grammars describing each intention. These grammars allow to tag each sentence belonging to one of the intentions of interest, and to extract for each sentence a set of relevant chunks which are then transformed into triples. These intentions are currently defined and used by Yseop's generation core engine. For the sake of the experiment, we choose to retain only simple intentions and sentences which can also be produced by this NLG engine.

2.2 Corpus transformation

2.2.1 Data logic

To ensure the precision and accuracy of the generated sentences, we have chosen a data-to-text representation method, which particularizes key elements of sentences in our financial corpus and extract triples, using an automated method close to Li et al. (2020).

This method was applied as to define a corpus of intentions. An intention is a sentence corresponding to a specific expression of a financial indicator's value. Yseop has shown that a handful of such intentions are sufficient to describe a data-driven narrative in a speciality domain, such as Finance. For

instance, an intention *DescribeValue* is a sentence stating the value of a financial indicator at a precise time and an intention *DescribeVariation* is a sentence describing the variation in time of a financial indicator's value. We use the prefix *Merge* to define a sentence composed of two or more intentions. In order to identify and extract these intentions in our corpus, a Ruta grammar (KLUEGL et al., 2016) was created to automate the triples extraction, imitating Gardent et al. (2017) data modeling. This grammar first uses dictionaries as well as POS-tag patterns to identify financial key elements related to these intentions and characterize them into one of the following categories:

- financial indicator
- reference (time and geographical element)
- measure
- predicate

Indicators, measures and dimensions are generic elements that can be found in all intentions. Predicates, on the other hand, vary according to the intention. To create a dictionary that take into account this specificity and can later be used for intention detection, we conducted a manual analysis of the market reports that allowed us to classify the predicates specific to each intention. Synonyms and antonyms have also been included to complete and enrich the dictionary (see Table 2 for some examples)

In a second stage, the grammar looks for syntactic combinations of these key elements. For example, a sentence containing exclusively a financial indicator, a state predicate, one measure and an optional time and/or geographic dimension will be extracted as an intention *DescribeValue*.

Sentences selected from financial corpus are then transformed into a set of triples (hereinafter referred to as **complete triples**), organized into subject-predicate-object structure, *&&* serving as a connector. See Table 11 in Appendix for a detailed overview.

2.2.2 Data construction

There is no theoretical limit to the maximum sequence a T5 can encode, the only constraint being the memory requirements. We did not work on this specific aspect as this is not the purpose of the experiment. We choose to work with a maximum input sequence of 400, trying to keep the experiments into a small memory consumption interval. Owing to the limited capability of our T5 model, triples that are too long cannot be fully processed by the model and therefore the generated sentences will be incomplete. In order to work around this problem, we trimmed the triples by replacing the elements with simpler ones and reducing the length of predicates, then creating simplified triples. In these simplified triples, financial indicators are replaced with their semantic class (predefined in our grammar). For example, *abuse tax* and *absolute organic operating costs* both belong to the class *expenses*, so they were replaced with the generic short form *expenses* in the simplified triple. Measures are replaced a simpler number ($ + two digits), all time dimensions are substituted by a preposition (if there is one in the initial dimension time) plus a year (*in 2019* , for example) and the expression *in America* replaces any term in the geography dimensions. We refer to this trimmed triples as **simple triples**.

We trained a model with simple triples to evaluate if our formalism was rich and accurate enough so the model could infer the data logic, and used the complete triples to train a model for inference.

According to the type and number of key components, target sentences are sorted into different intentions. Our grammar for now is able to recognize and annotate 8 intentions, *DescribeValue* and *DescribeVariation* being the core intentions, based on which we developed 6 others (see Table 1)

Two sets of experiments have been built for each of the data sets created from simple and complete triples. Each experiment is detailed in Section 5.

3 Data sets

Applying the triples generator on a 4159 raw corpus files, we have collected 20615 sentences annotated for both complete and simple triples. The frequency for each intention in each set is presented in Table 1.

We randomly sampled three different training and testing partitions in order to leverage the scarcity of our data. All measures provided below are aggregated means of these three partitions.

4 Models

We used the T5 sequence to sequence transformer from the transformers library by (Wolf et al. (2020)

Intention	Definition	# full data	# test data
DescribeValue	`Measure of an indicator`	4951	990
DescribeVariation	`Variation of an indicator`	7483	1492
DescribeValueWithContributor	`Measure of an indicator with contributing factors`	1294	250
DescribeVariationWithContributor	`Variation of an indicator with contributing factors`	304	61
MergeDescribeValue	`At least 2 DescribeValue`	5744	1149
MergeDescribeValueWithContributor	`At least 2 DescribeValue and one expression contributor`	74	15
MergeDescribeVariation	`At least 2 DescribeVariation`	729	146
MergeDescribeVariationWithContributor	`At least 2 DescribeVariation and one expression contributor`	36	7

Table 1: Complete list of Intentions

Infinitive	Semantics	Semantics +
`{grow}`	*describe object variation*	`{increase}`
`{record}`	*describe object state*	`{null}`
`{record an increase}`	*describe object variation*	`{increase}`
`{be higher than}`	*compare object*	`{above}`

Table 2: Examples of predicate dictionary entries

to run the experiments, using an Nvidia GeForce RTX 2070 GPU with 8 Go RAM.

We trained two models, one from complete triples, and another one from simple triples, for each of our 3 data partitions. A simple [1] and a complete [2] trained models are available for reproducibility on Hugging Face hub.

4.1 Training parameters

Our objective is to evaluate our data formalism and an associated sequence to sequence model capabilities, so we did not experiment much on fine-tuning. We used a standard set of training parameters for all models and trained for one epoch and batches of 6, using the Hugging Face transformers library and the AdaFactor optimization method, keeping all default parameters except for the following:

- learning rate lr=1e-3
- regularization constants eps=(1e-30, 1e-3)
- decay_rate=0.7

4.2 Generation parameters

At inference, we tried to limit hallucinations and omissions while maintaining a good level of richness on the structure and vocabulary of the generated sentences.

We use a mix of top_k _ and top_p sampling for generating. Top_k is a sampling scheme, in which the K most probable next tokens are filtered and the probability mass is redistributed among only those K next tokens. Top_p is also a sampling scheme, managing creativity of the model. It chooses from the smallest possible set of words whose cumulative probability exceeds the probability p. This way, the size of the set of words (a.k.a the number of words in the set) can dynamically increase and decrease according to the next word's probability distribution.

We chose the following process for selecting the most suitable top_p and top_k for our generator:

- select one representative sentence and its corresponding triples for every intention.
- prepare 10 top_p (from 0.12 to 1) and 10 top_k (from 10 to 100) and combine them in a pairwise fashion to get 100 (top_p, top_k) couples.
- generate 10 sentences from a single triple. Then measure the similarity between these 10 sentences with ROUGE [3] and collect the measure under different top_p and top_k couples. We considered this average ROUGE to measure the creativity of our model. The bigger it is (less variation in the 10 generated sentence), the less creative the model is.
- compare the 10 generated sentences with the initial sentence in order to collect the ROUGE measure under different top_p and top_k couples. The bigger it is, the more accurate our model is.

[1] https://huggingface.co/yseop/FNP_T5_D2T_simple

[2] https://huggingface.co/yseop/FNP_T5_D2T_complete

[3] https://github.com/pltrdy/rouge

The top_p and top_k selected for simple triples and complete triples are (0.72, 40) and (0.82, 90), respectively. The model gives the best performance with them, leading to results presented in section 6.

4.3 Evaluation metrics

During the experiment, we have noticed that both the length of elements in the triples and the model's familiarity with them can influence the quality of the generation. We have adopted 3 methods to assess the quality of our models.

- The generated sentences are compared with the initial sentences and the lexical similarity is measured with a BLEU score (Papineni et al. (2002)) [4], adapted so it considers bi-grams.
- The generated sentences are compared with the initial sentences and the semantic similarity is mesured with a BERT-SCORE (Zhang* et al. (2020).
- The generated sentences are reintroduced into the triples generator to obtain regenerated triples. The inspiration for regenerating the triples and evaluate them against the original ones comes from Veksler et al. (2019) 's work on how to assess a key level of information for NLG. The degrees of similarity between the regenerated triples and the original ones offers another point of view on the quality of generated sentences and assesses the credibility of the data logic initially chosen. We used both BLEU and BERT-SCORE to evaluate these similarities.We will refer to these measures as Triple BLEU and Triple BERT.

It is important to notice that the triples comparison results is fully automated and neither human evaluation nor inter-annotator agreement statistics have been performed. Since the triples production process biases the performance measure, and is used both at training and inference, we are in fact evaluating the capability of our model to preserve the "fixed-pointedness" of T5 with respect to our representation rather than the T5 natural language generation power.

5 Experiments

We defined two experiments, one training and evaluating for complete triples (see subsection 5.1), the other for simple triples (see subsection 5.2), and computed the four metrics previously detailed for each experiment. The measures provided are arithmetic means of the scores evaluated for all models created from our three different data partitions.

In the following subsections, we will refer to any element issued from the original corpus sentences as **original**. For each table of results, we present the actual number of triples that could be regenerated in regard to the number of sentences generated at inference available for testing.

5.1 Experiment 1

In this initial experiment, we used complete triples to fine-tune a T5 model. A data sample is available in Table 3, results are provided in Table 8. The BLEU score shows important variations in between intentions, due to the fact that some intentions are more complex and contain more elements than simpler ones like *DescribeValue*, and because they are less represented in the training data. Having around 4000 training examples seems to be a pre requisite to obtain significant improvement on the results.

5.2 Experiment 2

5.2.1 Simple

In this experiment we trained another model to learn and generate from simple triples.

The objective here is to workaround the limitations of our model in low memory consumption mode. The process for training a model for simple triples is the following:

- complete triples are simplified
- we simplify the original sentences by replacing the original elements by the simple ones (indexed by simple triples)

The model is trained with simple triples against these simplified sentences (see example provided in Table 4), then simplified original sentences used for training are compared with the sentences generated at inference (an evaluation sample is provided in Table 5).

5.2.2 Restored

To affect a metric to sentences generated at inference from simple triples models, we retain two additional features:

- in the sentences generated at inference, we restore the original elements using their index

[4] https://www.nltk.org/_modules/nltk/translate/bleu_score.html

in the original sentences, and compare these restored sentences with the original ones. An example of this transformation is provided in Table 6.

- we regenerate triples from these restored sentences, and compare them with the complete triples presented in section 5.1

We will refer to the triples and sentences in this experiment as **restored**. The results are provided in Table 10.

6 Results analysis

BLEU and BERT-SCORE leads to different conclusions and the BLEU score is generally lower than BERT-SCORE. This is because the 2 metrics evaluate the sentence at different levels.

6.1 Evaluating for triples

We expect sentences in financial report to contain all key information provided in the input data. However, BLEU and BERT-SCORE are incapable of examining the completeness of generated sentences. To achieve this goal, we passed the generated sentences to the triples generator and evaluate the regenerated triples with BLEU and BERT-SCORE. The higher the score is, the more complete the generated sentence is.

We were not able to regenerate any triple for a significant amount (22%) of test set sentences generated at inference, neither for complete triples nor for simple triples. The results take into account this information loss, when this happens the Triple BLEU and BERT-SCORE are evaluated to zero. This is partly due to our triples generator, partly to the structure of the sentence generated at inference time. Our triples generator is very dependant from the lexical layout of the sentence. In some cases, generated sentences which would be qualified for triple extractions are not recognized as such and ignored. On the other hand, and for the same reasons, the triple generator will also ignore ill-formed sentences.

The attribution to each case is still a work in progress. We provide examples of ignored generated sentences from which we could not regenerate any triple in Table 7.

This leads to important discrepancy in the Triple BLEU results, between different types of intentions and between complete versus simple triples experiments. Nevertheless we can still directly link the fidelity of the results to input data with the size of the training set.

Simple triples achieve significant better Triple BLEU score than complete triples, due to the fact that sentences generated from simple triples are shorter, and usually mirror the original simple triple sinformation much better than sentences generated from complete triples (often interrupted before a human readable sentence is fully generated at inference time, thus regenerating incomplete triples or none). For complete experiment, we were able to regenerate 85 % of the indicators present in the triples at inference and 95% of the indicators for the simple triples experiment.

6.2 Evaluating for sentences

BLEU evaluates the generated sentences on lexical level. The significant difference between complete and simple results comes mainly from the number of triples we were able to regenerate in each case, the simplest intentions for which a lot of training data was available being once again favored in both cases.

We can witness an improvement on average (from 0.423 average BLEU for sentences generated from complete triples at inference to 0.656 for sentences generated from simple triples), yet the similarity between restored sentences and original sentences (0.429 average BLEU) is only slightly higher than between original sentences and sentences generated from complete triples (0.423 average BLEU), mainly due to the risk of information loss during the process of restoring the original information in simplified sentences. .

Figure 1 shows that, as the complexity of intention increases, average BLEU score for simplified sentences generated from simple triples exceeds BLEU for complete sentences generated from complete triples and also restored generated sentences.

While the sentences are short (intention is less complex), a small lexical change (change of predicate for instance) is reflected in a big drop in BLEU score. For the same intention, the simplified generated sentence is usually the shortest. Therefore, under simpler intention, (e.g. *DescribeValue*), simplified sentences generated from simple triples obtained the lowest score. However, as the intention becomes more complex, the length of simplified sentences increases, which offsets the influence of lexical change in BLEU score. In addition, the BLEU score of simplified generated sentences re-

Triple	Generated Sentence	Regenerated Triple
Operating margin | valIs | 5.8% && 5.8% | comTo | 8.5%	Operating margin was 5.8% (versus 8.5%).	Operating margin | valIs | 5.8% && 5.8% | comTo | 8.5%

Table 3: Complete triples, generated sentence and regenerated complete triples example for original sentence *Operating margin was 5.8% compared to 8.5%.*

Original Triple	Generated Sentence	Regenerated Triple
Results | valIs | 10% && 10% | comTo | 11%	Results was 10% (-0.71) and remained at the same level compared with 11%	Results | valIs | 10% (-0.71) && 10% (-0.71) | comTo | 11%

Table 4: Simple triples, generated sentence and regenerated simple triples example for original sentence *Operating margin was 5.8% compared to 8.5%.*

Original Simplified sentence	Generated Sentence
Results was 10% compared to 11%.	Results was 10% (-0.71) and remained at the same level compared with 11%)

Table 5: Simplified original sentence and sentence generated from simple triples model at inference for *Operating margin was 5.8% compared to 8.5%.*

Restored sentence
Operating margin was 5.8% (-0.71) and remained at the same level compared with 8.5%

Table 6: Sentence generated from simple triples model at inference restored with original elements for *Operating margin was 5.8% compared to 8.5%.*

Original Sentence	Original Triple	Generated Sentence
Non-current liabilities were ¥309.0 billion, an increase of ¥2.4 billion or 0.8%, from the end of the previous fiscal year	Non-current liabilities | valIs | ¥309.0 billion && Non-current liabilities | incBy | ¥2.4 billion or 0.8%	Non-current liabilities were
The right-of-use asset and discounted lease liability related to discontinued operations are €398 million as at 1 January 2019.	Discontinued operations | infBy | the right-of-use asset && lease liability | valIs | €398 million && €398 million | dTime | as at 1 january 2019	The right-of-use asset for the right-of-use asset and the right-
For the nine months ended September 30, 2020, revenues were $421.7 million, up 8.9% or $34.4 million from $387.3 million in the same period in 2019.	Revenues | valIs | $421.7 million && $421.7 million | dTime | for the nine months ended September 30, 2020 && revenues | incBy | 8.9% or $34.4 million	Revenues for the nine months ended September 30, 2020 were $421.7 million, an increase of
Revenue for January-September period amounted to EUR 499.6 (400.5) million, an increase of 24.7%.	Revenue | valIs | eur 499.6 (400.5) && EUR 499.6 (400.5) | dTime | period && revenue | incBy | 24.7%	Revenue during the reporting period amounted to EUR 499.6 (400.5) million
Function costs were €13,266 million in 2018 (2017: € 12,790 million).	Function costs | valis | €13,266 million && €13,266 million | dTime | in 2018 && €13,266 million | comTo | € 12,790 million && € 12,790 million | dTime | 2017	Function costs amounted to €13,266 million in 2018 (2017: € 12,
Long-term Liabilities amount to EUR 5,479k (31 December 2016: EUR 6,866k).	Long-term liabilities | valIs | EUR 5,479k && EUR 5,479k | comTo | EUR 6,866k && EUR 6,866k | dTime | 31 december 2016	Long-term liabilities amount to EUR 5,479k (31 December 2016: EUR 6,866

Table 7: Non regenerated triples sample

Intention	# test	# nan triples	Triple BLEU	Triple BERT	Sentence BLEU	Sentence BERT
DescribeValue	990	163	**0.781**	**0.960**	**0.646**	**0.944**
DescribeVariation	1492	173	0.693	0.951	0.573	0.941
DescribeValueWithContributor	250	146	0.304	0.864	0.364	0.896
DescribeVariationWithContributor	61	14	0.443	0.908	0.325	0.911
MergeDescribeValue	1149	392	0.363	0.888	0.556	0.935
MergeDescribeValueWithContributor	15	6	0.264	0.868	0.346	0.904
MergeDescribeVariation	146	61	0.259	0.865	0.359	0.916
MergeDescribeVariationWithContributor	7	3	0.188	0.854	0.211	0.906
Mean	-	-	*0.412*	*0.895*	*0.423*	*0.919*

Table 8: BLEU and BERT-SCORE (F1) results by intention for complete triples model. *nan triples* stands for non regenerated triples

Intention	# test	# nan triples	Triple BLEU	Triple BERT	Sentence BLEU	Sentence BERT
DescribeValue	990	41	**0.941**	**0.990**	0.469	0.919
DescribeVariation	1492	55	0.914	0.985	0.590	**0.936**
DescribeValueWithContributor	250	93	0.499	0.899	0.407	0.889
DescribeVariationWithContributor	61	14	0.564	0.919	0.453	0.915
MergeDescribeValue	1149	110	0.819	0.964	**0.604**	0.931
MergeDescribeValueWithContributor	15	5	0.397	0.891	0.471	0.899
MergeDescribeVariation	146	27	0.656	0.934	0.491	0.919
MergeDescribeVariationWithContributor	7	2	0.459	0.890	0.405	0.899
Mean	-	-	*0.656*	*0.934*	*0.486*	*0.913*

Table 9: BLEU and BERT-SCORE (F1) results by intention for simple triples model. *nan triples* stands for non regenerated triples

Intention	Triples BLEU	Triple BERT	Sentence BLEU	Sentence BERT
DescribeValue	**0.823**	**0.971**	**0.554**	**0.936**
DescribeVariation	0.684	0.944	0.537	0.930
DescribeValueWithContributor	0.337	0.870	0.298	0.876
DescribeVariationWithContributor	0.351	0.884	0.323	0.897
MergeDescribeValue	0.653	0.936	0.588	0.932
MergeDescribeValueWithContributor	0.274	0.868	0.396	0.891
MergeDescribeVariation	0.494	0.903	0.434	0.911
MergeDescribeVariationWithContributor	0.257	0.859	0.299	0.891
Mean	*0.484*	*0.904*	*0.429*	*0.908*

Table 10: BLEU and BERT-SCORE (F1) results by intention for restored simple triples and sentences (# of test sentences and non regenerated triples is the same as in Table 9)

mains relatively steady compared to the other 2 types of generated sentences. This phenomenon tends to prove that simplification of initial sentences and initial triples does improve the performance. And another proof is that this method generates 1879 sentences with BLEU score in interval 0.98 to 1 for simple triples models, while the number of sentences generated from complete triples and restored triples models scoring within this interval is 1600 and 1783, respectively.

We evaluate with BERT-SCORE on semantic level. Taking BERT as the standard, there is little difference between the aggregated measures for sentences generated at inference from complete triples, simple triples or restored triples models. It's interesting to notice that restored simple sentences for well defined intentions such as *DescribeValue* exhibit a BERT-SCORE close to complete triples generated sentences (0.936 and 0.944 respectively), so this technique might be a way to workaround the memory constraints of the T5.

7 Error analysis

We have observed notable gaps between the BLEU and BERT-SCORE measures. We identified at least 4 reasons why this might occur:

1. Different verbs of same semantic meaning are employed in generated sentences:
    - **Original sentence**: The total gaming margin in online games during the quarter *amounted to* 4.7
    - **Generated sentence**: The total gaming margin in online games during the quarter *was* 4.7
2. The position of dimension time or dimension geography changes (slight influence):
    - **Original sentence**: Revenue *for 2016* amounted to 245 million.
    - **Generated sentence**: Revenue amounted to 245 million *for 2016*.
3. When the indicator in the triples starts with a lowercase, the model adds complement to it, which may be different from the complement in the original sentence. And sometimes, different complements may lead to different conjunctions of verb:
    - **Original sentence**: *The value of* deferred tax assets at 31 December 2016 *was* €190 million.
    - **Generated sentence**: *The total* deferred tax assets at 31 December 2016 *were* €190 million.
4. Predicates used in the triples don't indicate the tense of verbs. Hence, the tense of generated sentence may be different from the original one:
    - **Original sentence**: The annual savings in interest costs from this refinancing *amounts to* approximately US$29 million.
    - **Generated sentence**: The annual savings in interest costs from this refinancing *amounted to* approximately US$29 million.

The first three examples show that a lexical-based measure as BLEU is clearly not suitable to evaluate NLG systems.We tried to leverage this issue by evaluating triples against regenerated triples, this evaluation being less sensitive to semantic variations while retaining enough syntax for comparison.

8 Conclusion and future work

We have evaluated how a T5 sequence to sequence transformer behaves in data to text generation, using a combination of BLEU and BERT-SCORE on triples (with simple triples achieving the best Triple BLEU score of 0.656). The result gap between simple and complete triples experiments demonstrates that transforming initial sentences into simple ones and generating sentences from simple triples contributes to increasing the completeness of the generated sentences and the data logic accuracy.

Future work will focus on leveraging the induced bias of the triple generator as to propose more accurate automation of the triple extraction, and working on the current limitations of the T5 model to extend the length of input sequences keeping memory consumption as low as possible.

Acknowledgements

We thank the FNP 2021 Committee for the opportunity to publish this research experiment.

References

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG

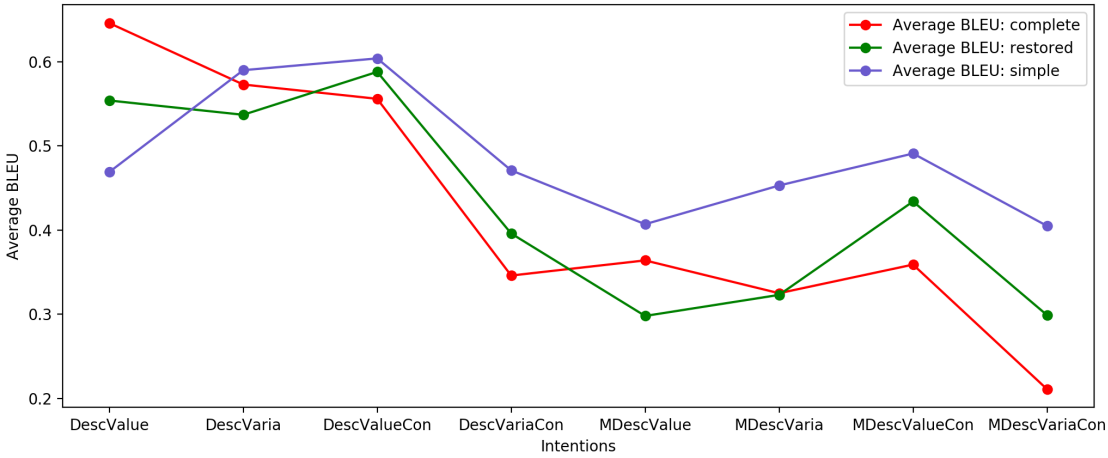

Figure 1: Average BLEU score on every intention for simplified generated sentence, complete generated sentences and restored generated sentences. From *DescribeValue* to *MergeDescribeVariationWithContributor*, the complexity of intention raises.

challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics.

Khushnuma Grover, Katinder Kaur, Karti Tiwari, and Kumar P. Rupali. 2021. Deep learning based question generation using t5 transformer. In *International Advanced Computing Conference (IACC 2020)*, volume 1367 of *Communications in Computer and Information Science*, Singapore. Springer. Https://doi.org/10.1007/978-981-16-0401-0_18.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

Zdenek Kasner and Ondrej Dusek. 2020. Data-to-text generation with iterative text editing. In *Proceedings of the 13th International Conference on Natural Language Generation, INLG 2020, Dublin, Ireland, December 15-18, 2020*, pages 60–67. Association for Computational Linguistics.

PETER KLUEGL, MARTIN TOEPFER, PHILIP-DANIEL BECK, GEORG FETTE, and FRANK PUPPE. 2016. Uima ruta: Rapid development of rule-based information extraction applications. *Natural Language Engineering*, 22(1):1–40.

Ziran Li, Zibo Lin, Ning Ding, Hai-Tao Zheng, and Ying Shen. 2020. Triple-to-text generation with an anchor-to-prototype framework. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3780–3786. International Joint Conferences on Artificial Intelligence Organization. Main track.

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*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Raphael Salmon. 2017. *Natural language generation using abstract categorial grammars*. Ph.D. thesis, Sorbonne Paris Cité.

Yael Veksler, Natalia Vanetik, and Marina Litvak. 2019. EASY-M: Evaluation system for multilingual summarizers. In *Proceedings of the Workshop MultiLing 2019: Summarization Across Languages, Genres and Sources*, pages 53–62, Varna, Bulgaria. INCOMA Ltd.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Appendix

Predicates	General form	Example
valIs	Indicator | valIs | Measure	Net cash inflow was 9 067 million USD. Net cash inflow | valIs | 9 067 million USD
valIs	Indicator | valIs | Measure	Net cash inflow was 9 067 million USD. Net cash inflow | valIs | 9 067 million USD
chaBy	Indicator | chaBy | Measure	The Company recorded a change of €12. net cash inflow | chaBy | €12
decBy	Indicator | decBy | Measure	Net cash inflow decreased by 12%. Net cash inflow | decBy | 12%
decTo	Indicator | decBy | Measure	Net cash inflow decreased to €10 million. Net cash inflow | decTo | €10 million
dFrom	Indicator | dFrom | Measure	Net cash inflow decreased from €12 million. Net cash inflow | dFrom | €12 million
incBy	Indicator | incBy | Measure	Net cash inflow increased by 12%. Net cash inflow | incBy | 12%
incTo	Indicator | incTo | Measure	Net cash inflow increased to €10 million. Net cash inflow | incTo | €10 million
iFrom	Indicator | iFrom | Measure	Net cash inflow increased from €9 million. Net cash inflow | iFrom | €9 million
Contribute	Indicator | Contribute | value contributed	Cash and cash equivalent was €20 million, with net cash inflow of €12 million. net cash inflow | Contribute | €12 million
CauBy	Indicator in result | CauBy | reason	Due to higher costs in services, costs increased by US$ 4 million. costs | CauBy | higher costs in services
InfBy	Indicator | InfBy | related factor	Full-year capital expenditure amounted to €24.2 million, mainly relating to new finishing capacity Full-year capital expenditure | InfBy | new finishing capacity
ContriBy	Contributed | ContriBy | Contributor	Cash and cash equivalent was €20 million, with net cash inflow of €12 million. Cash and cash equivalent | contriBy | net cash inflow
dTime	Measure | dTime | Date	Cash and cash equivalent was €20 million in 2019. €20 million | dTime | in 2019
startDate	startValue | startDate | Date	The revenue increased from €20 million in 2019 to €23 million in 2020. €20 million | startDate | in 2019
endDate	endValue | endDate | Date	The revenue increased from €20 million in 2019 to €23 million in 2020. €23 million | endDate | in 2020
diGeo	Measure | diGeo | Dimension geography	The revenue increased by €20 million in Europe. €20 million | diGeo | in Europe
cTime	Measure | cTime | Date for comparison	The revenue increased by €20 million compare to the prior year. €20 million | cTime | the prior year
comTo	Measure | comTo | Measure for comparison	The revenue was €20 million (in 2019: €21 million) compare to the prior year. €20 million | comTo | €21 million

Table 11: Usage of predicates