HPSG/MRS-Based Natural Language Generation Using Transformer

Gyu-min Lee
Department of Linguistics
Korea University
gyuminlee@korea.ac.kr

Sanghoun Song
Department of Linguistics
Korea University
sanghoun@korea.ac.kr

Abstract

With the massive success of stochastic language models, some claim that the symbolic approach is not necessary for natural language processing. However, recent studies made use of linguistically motivated meaning representation of Abstract Meaning Representation and Minimal Recursion Semantics in a natural language generation task and achieved impressive results while hinting that neural networks do make extensive use of linguistic information. This study partially replicates previous studies with Transformer. The model performed worse than the previous researches. We present some evidence that the model failed to make the correct lexical choices due to Attention weight assignment. Overall, it appears that models built solely on Attention mechanism are suboptimal for processing language representation with rich grammatical information, while hinting the possible use of engineered grammar for improving the controllability of the neural models.

1 Introduction

While symbolic natural language processing (NLP) models have great precision, they have limited coverage. Meanwhile, stochastic NLP models, notably neural NLP, boast broad coverage but lack controllability. Then, is it possible to join the two frameworks and improve a model's capacity, precision, and controllability at the same time? The current study explores a way to integrate the two approaches by replicating previous researches with Transformer (Vaswani et al., 2017) and finds out that Attention-dependant approaches may not be optimal for processing grammatical representations.

Modern NLP is largely stochastic. From statistical language processors to the neural language models, NLP rapidly moved from its symbolic, algorithm based predecessors. When it comes to the empirical evaluation, the stochastic NLP models are performing much better while being cheaper to design conceptually. In fact, stochastic NLP models perform so well that some question the necessity of symbolic knowledge of linguistics itself in NLP.

However, neural language models inevitably suffer from the "black box" problem. In other words, we do not really know how they work. Therefore, debugging the neural language models and make them perfect is a difficult process. Some researchers solve this problem by increasing model sizes. However, its environmental impact cannot be neglected, needless to say whether it is ever possible for such gigantic language models can indeed "learn" the language (Bender et al., 2021).

On the other side of the coin, it is possible to engineer a computational grammar out of certain linguistic formalism. Called grammar engineering (Bender and Emerson, 2021; Bender et al., 2008), this process bore several computational grammars including English Resource Grammar (ERG; Flickinger, 2000), which is an English grammar based on the Head-Driven Phrase Structure Grammar (HPSG; Pollard and Sag, 1994) framework. While such grammars show high precision, they show low recall (Bender and Emerson, 2021), indicating that their coverage is limited.

Hajdik et al. (2019) integrates neural natural lan-

guage generation (NLG) with ERG. In detail, they defined NLG as a task of translating from semantic representation rich in linguistic details to natural language sentence. In this machine translation (MT) task, they achieved a high BLEU score (Papineni et al., 2002), showing that neural sequence-to-sequence model consisting of bidirectional Long Short-Term Memories (LSTMs; Hochreiter and Schmidhuber, 1997) and Attention mechanism (Bahdanau et al., 2014) can faithfully translate MRS representation into natural language text.

The current study replicates Hajdik et al. (2019) with Transformer. MRS representation is long. Therefore, applying a recurrent neural network (RNN) like LSTM inevitably has limitation due to the vanishing gradient problem. By applying Transformer, we expected to further improve the performance of the model of Hajdik et al. (2019). However, it turned out that the Transformer model is actually underperforming. It appeared that the Transformer model failed to draw correct lexical choices from MRS representation.

The current paper is structured as follows. Section 2 details the modern stochastic NLP and grammar engineering with their strength and weakness, along with the description on the RNNs and the vanishing gradient problem. Section 3 gives out the detail of the data, model, implementation method, and evaluation scheme. Section 4 contains the quantitative and qualitative results of our model. In Section 5, we discuss the significance of our research. Section 6 summarizes and concludes the paper.

2 Background

2.1 Stochastic Natural Language Processing

The introduction of machine learning techniques such as neural networks brought massive improvement to the field of NLP. From N-Gram language models to Transformer based language models like BERT (Devlin et al., 2018), NLP models developed rapidly away from the rule-based models. Such stochastic models are so powerful that their performance is often compared to human abilities.

Modern high-performing neural language models share a commonality. That is, they are not based on linguistic motivation. BERT, for instance, is not built upon a linguistic theory. The learning objectives of BERT, namely masked language model and next sentence prediction, are built purely on a processing basis and have arbitrary nothing to do with linguistic theories.

Recent studies on BERT and other language models, namely the BERTology (Rogers et al., 2020), found that neural language models learned some syntactic information based particularly on their high performance on long distance dependency constructions. In other words, while traditional statistical stochastic models such as N-gram language models only captured the linear pattern in the corpora, neural language models seem to be able to learn the hierarchical structure of language.

To this end, some claim that natural language texts can be processed and generated successfully only with the text information alone. A deep learning textbook (Subramanian, 2018), for instance, introduces NLP as a field with no need for feature engineering. In other words, it claims that raw text can be processed only with raw text alone without any domain knowledge like linguistics.

While such neural language models boast in their empirical performance, their mechanisms are not transparent. In other words, we cannot look into the neural models clearly. Often referred to as a "black box" problem, this inability to take a clear look into the model makes it extremely challenging to debug and improve the models. Moreover, it is difficult to control the model to produce certain output.

2.2 Grammar Engineering

Grammar engineering is to build a computational and machine-readable representation of the formalism of a grammatical framework (Bender and Emerson, 2021; Bender et al., 2008; Song, 2014). While many grammatical frameworks have been used for grammar engineering, ERG made use of HPSG framework. According to Bender and Emerson (2021), HPSG is suitable for grammar engineering as its theory is independent of its formalism.

Such engineered grammars can both parse and generate natural language sentences. For example, ERG can parse a sentence into a meaning representation of MRS. Since HPSG rules are defined bidirectionally, ERG can also generate natural language sentences from MRS representation (Bender and Emerson, 2021). Using this property, ERG can be used for

a variety of tasks. For instance, Bond et al. (2008) used ERG to paraphrase English sentences in a parallel corpus to improve the statistical MT performance.

However, despite its high precision, engineered grammars such as ERG suffer from low recall (Bender and Emerson, 2021). In other words, their coverage is limited. In contrast, neural language models can process arbitrary any sentences. In other words, compared to neural language models, engineered grammars are limited to sentences that can be explained with the grammatical framework and formalism they are based on. For neural language models have broad coverage and acceptable performance while being relatively cheaper to build conceptually, machine learning has quickly become a dominant method of doing NLP.

2.3 Linguistic Meaning Representation

NLG is a task of generating human language sentences out of either linguistic sequence of tokens (*text-to-text generation*) or non-linguistic pieces of information (*data-to-text generation*) (Gatt and Krahmer, 2018; Yu et al., 2019; Puzikov et al., 2019). For instance, summarization is a text-to-text generation task as it takes an existing text, or the linguistic sequence of tokens, as input. On the other hand, in data-to-text generation, a meaning representation such as a table is used to generate sentences.

Konstas et al. (2017) redefined data-to-text NLG task as an end-to-end MT task from a linguistic meaning representation, namely abstract meaning representation (AMR). In other words, they built an MT model that translates AMR into natural language text using a sequence-to-sequence model with bidirectional LSTMs and Attention mechanism.

While AMR formalism abstracts away the surface form to represent human language sentences as a directed graph, other formalisms contain richer information. MRS, for instance, contains richer information about the surface form, including number, tense, and case (Konstas et al., 2017). Using this richer representation, Hajdik et al. (2019) reproduced the work of Konstas et al. (2017) and showed a significant improvement in translation performance when measured with BLEU metric. In this process, ERG was used to get the MRS representation for the training data.

2.4 RNNs and Their Limits

Like an animal neuron, a perceptron takes multiple inputs, applies an activation function to the sum of the inputs, and provides a single output. In turn, neural networks take an input, process it with multi-layered perceptrons, and provide a single output. Since the multi-layered perceptrons are hidden to the users, it is called a hidden layer. Compared to simple neural networks, RNNs are specialized to learn from sequential data by making the hidden state accessible to the next step. In other words, their hidden states can refer back to their past hidden states, whereas in other architectures, the next step can only refer to the output of the previous step.

Such neural networks learn from the data by measuring the loss of their output and back-propagate the loss through hidden states to update the parameters of their perceptrons. RNNs' ability to learn from a sequence of data comes from the fact that the hidden states are referable. Unlike other networks, the loss is back-propagated through the time steps. Therefore, RNNs are able to learn from a sequence of data efficiently. However, in this process, due to the derivative calculus used in back-propagation, the further time steps do not really learn from the current time step as their calculus result converges to 0.

Known as the *vanishing gradient problem*, this issue has made engineers come up with newer models such as LSTM and Gated Recurrent Unit (Cho et al., 2014). Such models tried to keep the important information longer by strategically "forgetting" the information considered unimportant. While such models were more effective than pure RNNs in processing longer sequences, they could not ultimately solve the vanishing gradient problem. RNNs and their variants inherently have hardship in processing longer sequences.

On the other hand, Attention mechanism, along with sequence-to-sequence, or encoder-decoder framework, was proposed to help the model to pay attention to the relevant part of the sequence regardless of its length. Built solely upon this mechanism, Transformer performed better at processing longer sequences, particularly for MT tasks. Specifically, Transformer utilizes Attention mechanisms only, which configure dependencies regardless of distance for more parallelization capability and per-

formance (Vaswani et al., 2017). Since MRS representation, linearized as in Hajdik et al. (2019), tends to be long, it is expected that applying Transformer would further improve the performance in this NLG task.

3 Method

3.1 Data

To compare our results with the previous research of Hajdik et al. (2019), we made use of the data provided by their work. The data consist of gold and silver datasets prepared with HPSG motivation. The gold dataset is the Redwoods Treebank (Oepen et al., 2004) (release 1214). The Redwoods Treebank is a parallel corpus of natural language sentences and their MRS representations. The MRS representations here were predicted using ERG. The results were manually checked by human reviewers. Thus this dataset is assumed to be a parallel corpus of correct MRS representations.

The silver dataset consists of a million sentences from the Gigaword Corpus. The sentences were selected to match with the domain and time period of the gold dataset. Hajdik et al. (2019) used ERG with ACE processor to generate their MRS representations.[1] While ERG can generate valid MRS representations, it is not manually checked. Therefore, one cannot guarantee whether the MRS representations in this dataset are correct.

Hajdik et al. (2019) showed that using silver dataset is highly beneficial to the model's performance through an ablation study. However, in this experiment, we used only gold dataset. By using gold dataset only, we perform the pilot study before training a model with a massive amount of data. Also, we aim to test the implementation of the model. Lastly, we aim to better understand the model's behavior with smaller dataset. The dataset, therefore, is 82,802 MRS-sentence pairs, divided into 67,313, 5,288, and 10,201 for train, validation, and test sets.

It is neither practical nor effective to feed the model with the multilinear MRS representation as shown in #1 of Figure 1. Therefore, we need to linearize the MRS representation. In Konstas et al. (2017), PENMAN format was utilized to linearize a directed graph representation of AMR. Dependency MRS (DMRS; Copestake, 2009) is a representation of MRS with a directed graph, and it is interchangeable with MRS representation. Therefore, the MRS representations were converted into DMRS, then into PENMAN (#2), following Goodman (2018). The PENMAN was linearized into a single line string (#3) following Konstas et al. (2017).

The data were then anonymized according to ERG's named entity recognition by replacing the tokens on the raw text to reduce data sparsity. During the entire process, the NLTK implementation of Moses tokenizer (Bird et al., 2009) was used. The entire data preparation process was done locally using the data and code provided by Hajdik et al. (2019).

3.2 Model and Implementation

As previously mentioned, Konstas et al. (2017) and Hajdik et al. (2019) employed bidirectional LSTMs for their sequence-to-sequence model along with the Attention mechanism. In contrast, the current work applies Transformer. Transformer performs better than traditional RNNs with longer texts, offering a breakthrough to MT systems particularly. Since linearized MRS representation can be considerably long, we predicted that we could increase the performance of the model from Hajdik et al. (2019) by adopting Transformer.

The model was implemented using OpenNMT-py (Klein et al., 2017).[2] For the hyperparameters, we used the suggested parameters from the OpenNMT-py documentation, which mimic the original setup from Vaswani et al. (2017).[3] Since Transformer is highly sensitive to hyperparameter settings, they were not changed except for the validation, which was changed from every 10,000 steps to every 5,000 steps.

The training was carried out using Google Colab. This free-to-start web application from Google provides access to a robust GPU environment in the cloud but limits the execution up to 12 hours. Therefore, the models were frequently saved, upon every 5,000 steps specifically, and when the operation limit was reached, an OpenNMT-py feature of training

[1] http://moin.delph-in.net/wiki/AceTop

[2] Specifically, release v.2.0.0.rc2 was used.

[3] https://opennmt.net/OpenNMT-py/FAQ.html#how-do-i-use-the-transformer-model

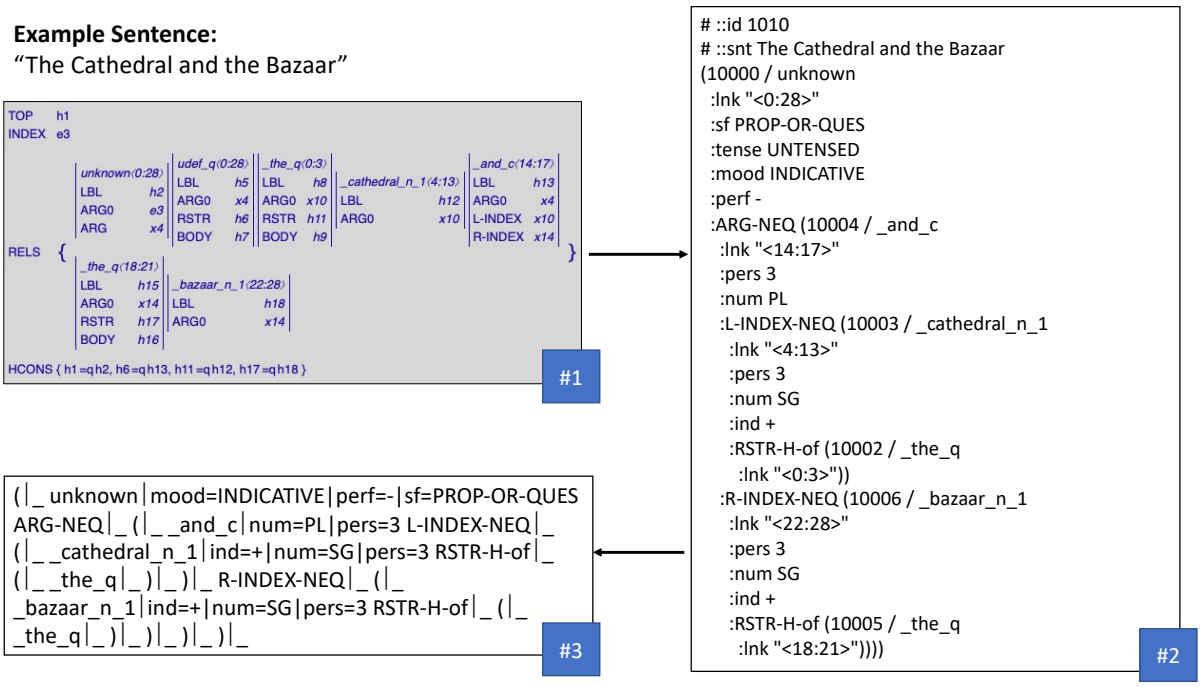

Figure 1: MRS linearization process

from a saved model was used to continue training from the latest checkpoint. The training was carried out up to 70,000 steps. During the training, a single Tesla T4 GPU was used.

3.3 Evaluation

BLEU is a metric for the automatic evaluation of machine translation. The metric itself is not designed for NLG. However, as Konstas et al. (2017) and Hajdik et al. (2019) realized NLG systems based on an MT approach, they used this automatic metric to evaluate their NLG systems. The current research also utilizes BLEU to evaluate the objective performance of the model.

To measure the BLEU score, the test dataset was translated using the saved models. The translation was detokenized and deanonymized following Hajdik et al. (2019). Automatic calculation of BLEU was carried out using SACREBLEU (Post, 2018) for more comprehensive results and for comparison with Hajdik et al. (2019). During the translation, beam search with the beam width of 5 was used.

Model	BLEU
Konstas et al. (2017)	33.8
Hajdik et al. (2019)	77.17
Hajdik et al. (2019) (gold only)	66.11
Ours	64.2

Table 1: Comparison of the results

4 Results

4.1 The BLEU Score

Konstas et al. (2017) achieved up to 33.8 BLEU. Hajdik et al. (2019) significantly improved the result to 77.17 for the entire dataset, and 83.37 for partial dataset. For their experiment that used only gold dataset, they achieved up to 66.11 BLEU for the entire gold dataset.

While Transformer was expected to further improve the result by resolving the vanishing gradients issue, the result turned out to be the opposite. After running the Transformer for up to 70,000 steps, we measured the BLEU score for the models that were saved every 5,000 steps. The score peaked at 30,000

steps with 64.2 BLEU. The score decreased afterward with the accuracy, perplexity, and cross entropy plateauing, hinting the overfitting of the model after 30,000 steps.

4.2 Translation Samples

(1) a. **PREDICTION**: If I am correct, they will help you understand exactly what it is saying the Linux community of good software - and perhaps they will help you become more productive yourself.

b. **ANSWER**: If I'm correct, they'll help you understand exactly what it is that makes the Linux community such a fountain of good software—and, perhaps, they will help you become more productive yourself.

The actual translation samples reveal an additional problem with using Transformer for MRS representations. For the PREDICTION, we present a detokenized and deanonymized prediction by the model. This corresponds to the model's translation from the MRS representation. The ANSWER is the original text the model is supposed to translate to. Overall, the Transformer model seems to perform reasonably well with relatively longer MRS representations, as in (1).[4]

(2) a. **PREDICTION**: The myth and the sword.

b. **ANSWER**: The Cathedral and the Bazaar

```
(|_ unknown|mood=INDICATIVE|perf
   =-|sf=PROP-OR-QUES ARG-NEQ|_ (|
   _ _and_c|num=PL|pers=3 L-INDEX-
   NEQ|_ (|_ _cathedral_n_1|ind=+|
   num=SG|pers=3 RSTR-H-of|_ (|_
   _the_q|_ )|_ )|_ R-INDEX-NEQ|_
   (|_ _bazaar_n_1|ind=+|num=SG|
   pers=3 RSTR-H-of|_ (|_ _the_q|_
    )|_ )|_ )|_ )|_
```

However, it turns out, the model struggled with the seemingly easy task of lexical choice. For instance, the linearized MRS given above seems straightforward. The lexical items of answer (2b) are all given in the MRS representation. However, as can be seen with (2a), it appears the model failed to catch the lexical item and chose different item in its vocabulary instead.

(3) a. **PREDICTION**: = = = Objectives = = =

b. **ANSWER**: Abstract

```
(|_ unknown|mood=indicative|perf
   =-|sf=prop-or-ques arg-neq|_ (|
   _ _abstract_n_1|ind=+|num=sg|
   pers=3 )|_ )|_
```

The trend continues even with a single word sequence (3). While the linearized MRS contains only a single word, *abstract*, as given above, the model answered with *Objectives*. This trend persists throughout the prediction: while the syntactic structure appears to be translated relatively well, it appears Transformer model failed to make correct lexical decisions.

4.3 Error Analysis

In order to understand the reasons for the lower performance of this model, we manually inspected 100 randomly selected translation samples. In detail, we compare the anonymized and undetokenized predictions from the best-performing 30,000 step model. Each translation was tagged with: no error, lexical choice error, syntactic error, punctuation error, and missing elements error. Since some of the grammatical information can be abstracted away even with MRS, some differences were not counted as errors. Those differences are the location of adverbial phrases that do not alter the meaning (see (4)), the use of aspect (present on behalf of present progressive, or vice versa, like (5)), use of clitics (*will* on behalf of *'ll*), and unreasonable punctuations (sentences that end with a quotation mark without opening quotation mark, like (5)).

(4) a. **PREDICTION**: do you want to travel around what time ?

b. **TARGET**: around what time do you want to travel ?

(5) a. **PREDICTION**: What am I doing now ?

b. **TARGET**: What do I do now " ?

[4] We used the best-performing 30,000 step model for all the predictions here.

Error	Number	Sample Prediction
No Error	47	*Okay , we have card0 options .*
Lexical	31	*I assume there is a full* ***salon*** *on the shipping costs .*
Punctuation	8	***: **** *named0*
Lexical & Missing Argument	5	***Don 't Linger***
Lexical & Syntactic	4	*When ad dollars is tight , the high page cost is* ***generally*** *a major* ***UNKcontributor0*** *for UNKadvertisers0 who want to appear regularly in a publication or not at all .*
Missing Argument	3	*Requesting immediately .*
Syntactic	2	***polite0 refund .***
SUM	100	

Table 2: Number of errors from the 100 translation samples. The errors in the sample prediction are marked in bold face.

As Table 2 summarizes, around half of the translations presented no error. This trend coincides with Hajdik et al. (2019), in which the manual inspection of the result showed that BLEU metric was underestimating the model due to issues such as formatting. It appears then that, while the model was generally able to generate acceptable sentences from linearized MRS representation, the details that are not reflected in the representation prevented the model from achieving a high score.

Among the 53 erroneous cases, 40 cases involved lexical choice problems similar to (2) and (3). This supports our assumption that the model learned to translate the syntactic aspect of MRS representation fairly well, but failed at making correct lexical choices from it. We assume that the issue arises from the Attention mechanism, on which Transformer is built. An MRS representation contains many functional keywords and symbols while containing few lexical tokens inside. Thus, it appears that the Attention mechanism pays attention to functional keywords instead of lexical items, thus failing to make a correct lexical decision.

To further investigate this assumption, we retrieved Attention weights from the translation of (2) using the `attn_debug` option of OpenNMT-py. There are some concerns about this method of proving the model by Attention weight. For instance, Serrano and Smith (2019) suggests that the Attention weight is not a reliable indicator. Brunner et al. (2019) mathematically reviews the Transformer model and suggests that attention weights are not directly interpretable. Moreover, Pruthi et al. (2019) demonstrates how easy it is to manipulate the Attention weights. In summary, there is a view that "Attention is Not Explanation" (Jain and Wallace, 2019).

However, Wiegreffe and Pinter (2019), as a response to Jain and Wallace (2019) specifically, claims that the explanatory ability of Attention mechanism depends on the definition of *explanation*. In their defense of the use of Attention weights as explanation, they decompose the explanation of an artificial intelligence into transparency, explainability, and interpretability, and empirically prove that the Attention mechanism does provide transparency to the model, providing a way to look into it. In sum, Wiegreffe and Pinter (2019) points out that while Attention model does not necessarily give faithful explanations, it still offers a plausible explanation.

As can be seen above, there is a debate on the validity of using the Attention weight for the explanation of a model. The debate is largely ongoing, but it appears to arise mainly from the ill-definition of the term "explanation". The previous works suggest that Attention mechanism still can be seen as a plausible way to investigate the model. Therefore, we report the result of the Attention weight, not to give a definitive explanation on our model's inability to focus on the lexical items, but as a piece of evidence pointing toward that direction.

The Attention weights returned by OpenNMT-py were visualized in a heatmap using the Seaborn package for Python. Here, the original sentence is presented on the horizontal axis, and the prediction to-

kens on the vertical axis. A brighter color indicates stronger attention. The results (Figure 2) show that when the tokens of *myth* and *sword* were both expected to pay the strongest attention to the lexical items of *cathedral* and *bazaar*, they were instead paying attention to the functional items that indicate the syntactic positions of the items.

5 Discussion

The present study leads to two findings. First, it is borne out that computational grammars such as ERG do help neural NLG. Following Konstas et al. (2017) and Hajdik et al. (2019), this research still demonstrates that a neural model can faithfully generate sentences in terms of syntax from rich semantic representation of MRS. Second, representations analyzed with a grammar should be handled with care, particularly with modern Attention-based approaches. Unlike natural language texts, grammar-represented texts come with many annotation symbols and few lexical items. This can cause the Attention mechanism to pay less attention to the lexical items embedded in the grammar representation, as in this study.

One may question the need for this study. The model of the current study is underperforming even than the engineered grammar of ERG in that it is unable to draw correct lexical decisions from the MRS representations. Indeed, ERG is a robust system by itself, and it can precisely and strictly generate English sentences (Bender and Emerson, 2021; Bond et al., 2008). However, the current study aims to generate a system that has more recall than ERG. As aforementioned, engineered grammars lack in its coverage. Meanwhile, deep learning techniques has a much broader coverage than engineered grammars. By joining the ERG with deep learning technology, we expect to achieve a model with high precision *and* recall.

By such hybrid models, we expect to make a deep learning system that is more controllable. Concretely, by having a neural generation system that translates MRS representation to natural language sentence perfectly, we would be able to control the generation process by precisely adjusting the MRS representation. In other words, we expect to build a controllable NLG model that is also flexible. While numerous NLG models have been suggested with neural approach, in practice, template-based models are still in use in commercial context (Dale, 2019; Mahamood and Zembrzuski, 2019) partially as it is difficult to control the neural NLG model – while neural NLG models can generate text, it is difficult to ensure the reliability of such models. We hope that the integration of MRS representation can bring that reliability to the neural generation model.

6 Conclusion

The current research partially reproduced Hajdik et al. (2019) with Transformer. Hajdik et al. showed that HPSG-based computational grammar, such as ERG, can improve neural NLG. This research takes that insight and applies a newer mechanism, expecting that it would perform better at this specific task by better processing longer sequences.

However, the results of the current study show that the Transformer model struggles with this MT task. It appears that the model failed to extract lexical items from linearized MRS appropriately. On the other hand, it could retrieve syntactic structure from it. The full investigation of this is beyond the scope of the current research. However, we suspect that relying on Attention mechanism alone is suboptimal for the model to interpret the linearized MRS representation.

MRS is a concise representation of the syntactic and semantic information of a sentence. In other words, each item of linearized MRS contains essential information that determines the correct surface form. By using Transformer, it appears that our model paid attention to the syntactic information of the linearized MRS, but not to the individual lexical items. When the model pays attention to a part of the input sentence, it can be interpreted that the other parts are neglected. This ignorance of the lexical items thus resulted in the poorer performance with Transformer even when the task was an MT, where Transformer generally shows better results.

Currently, we are working on a similar experiment with silver dataset as well. For future research, we plan to further probe into our model using other test sets to investigate its ability, along with comparing the results with models of Hajdik et al. (2019) to evaluate how beneficial Transformer model is in terms of extracting syntax. Also, considering Attention mechanism's internal disadvantage of ignoring lex-

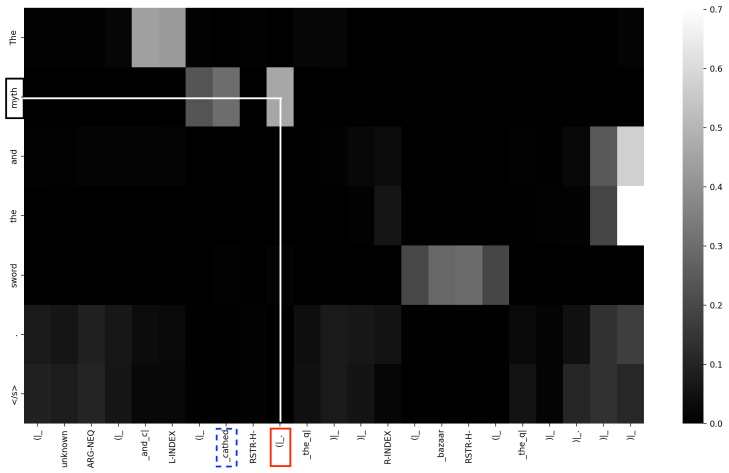

Figure 2: The Attention weight for (2) visualized as a heatmap. *myth* payed most of its attention to a functional item (the red, lined box), rather than the lexical word of *cathedral* (the blue, dotted box). Brighter color means higher attention weight. Lines and boxes are added by the authors for illustration.

ical items, other advanced RNNs that model hierarchical information explicitly and effectively can be applied. Finally, one may adjust the Attention mechanism so that it can pay more attention to lexical items even when they are surrounded by grammar information symbols.

References

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

Emily M Bender and Guy Emerson. 2021. *Computational linguistics and grammar engineering*, prepublished version edition. Berlin: Language Science Press.

Emily M Bender, Dan Flickinger, and Stephan Oepen. 2008. Grammar engineering for linguistic hypothesis testing. In *Proceedings of the Texas Linguistics Society X Conference: Computational linguistics for less-studied languages*, pages 16–36. Citeseer.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.

Francis Bond, Eric Nichols, Darren Scott Appling, and Michael Paul. 2008. Improving statistical machine translation by paraphrasing the training data. In *International Workshop on Spoken Language Translation (IWSLT) 2008*.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2019. On identifiability in transformers. *arXiv preprint arXiv:1908.04211*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Ann Copestake. 2009. Slacker semantics: Why superficiality, dependency and avoidance of commitment can be the right way to go. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 1–9.

Robert Dale. 2019. Nlp commercialisation in the last 25 years. *Natural Language Engineering*, 25(3):419–426.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dan Flickinger. 2000. On building a more effcient grammar by exploiting types. *Natural Language Engineering*, 6(1):15–28.

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.

Michael Wayne Goodman. 2018. *Semantic operations for transfer-based machine translation*. Ph.D. thesis, University of Washington.

Valerie Hajdik, Jan Buys, Michael W Goodman, and Emily M Bender. 2019. Neural text generation from rich semantic representations. *arXiv preprint arXiv:1904.11564*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. *arXiv preprint arXiv:1902.10186*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. *arXiv preprint arXiv:1704.08381*.

Saad Mahamood and Maciej Zembrzuski. 2019. Hotel scribe: Generating high variation hotel descriptions. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 391–396.

Stephan Oepen, Dan Flickinger, Kristina Toutanova, and Christopher D Manning. 2004. Lingo redwoods. *Research on Language and Computation*, 2(4):575–596.

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 on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Carl Pollard and Ivan A Sag. 1994. *Head-driven phrase structure grammar*. University of Chicago Press.

Matt Post. 2018. A call for clarity in reporting bleu scores. *arXiv preprint arXiv:1804.08771*.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C Lipton. 2019. Learning to deceive with attention-based explanations. *arXiv preprint arXiv:1909.07913*.

Yevgeniy Puzikov, Claire Gardent, Ido Dagan, and Iryna Gurevych. 2019. Revisiting the binary linearization technique for surface realization. In *Proceedings of The 12th International Conference on Natural Language Generation*, pages 268–278.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *arXiv preprint arXiv:2002.12327*.

Sofia Serrano and Noah A Smith. 2019. Is attention interpretable? *arXiv preprint arXiv:1906.03731*.

Sanghoun Song. 2014. *A grammar library for information structure*. Ph.D. thesis, University of Washington.

Vishnu Subramanian. 2018. *Deep Learning with PyTorch: A practical approach to building neural network models using PyTorch*. Packt Publishing Ltd.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. *arXiv preprint arXiv:1908.04626*.

Xiang Yu, Agnieszka Falenska, Ngoc Thang Vu, and Jonas Kuhn. 2019. Head-first linearization with tree-structured representation. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 279–289.