What Does Image Say About Syntactic Representation: An Investigation of Visually Grounded Neural Syntax Acquisition

Zijiao Yang, Xiaoliu Chen, Yuping Li

University of Colorado, Boulder

Abstract

This project investigates the Visually Grounded Neural Syntax Acquisition (VG-NSL) model by Shi et al. (2019). VG-NSL is an unsupervised model that learns syntactic representation and structures by looking at images and reading paired captions. We replicated the major results of the paper and found that VG-NSL model indeed shows overall higher F1 score compared with baseline models, is more robust under different initialization and is more data efficient. Meanwhile, we modified the model with different reward functions, investigated the possible reasons behind the model's data efficiency and explored if BERT word embedding is a feasible way to improve the model.

1 Introduction

Natural language processing has been flourishingly advanced in the past few years, various benchmark tasks have reached human baselines, for example, Recognize Textual Entailment and Sentiment Analysis has come to an accuracy score of over 90%. However, a recent analysis has shown that in spite of the strong benchmark task result, current natural language models are still merely doing pattern recognition of the associations written explicitly in text.

Human understands language by grounding them into a rich context of the physical world, with multi-modal information available, and learns natural language simultaneously with how the world works. An interesting question then is what role does multi-modal information plays in natural language understanding for human beings, can we extract intuitions from it to build better natural language systems?

To explore answers to the above questions, in this project, we focus on the interaction between two modality data: text and image, to investigate if visual information can be integrated with textual information to contribute to language understanding. Our project is based on the paper titled "Visually Grounded Neural Syntax Acquisition" by Shi et al., 2019. We replicated the major results of this paper and did some exploration and extensions in the hope of better understand how an integration of text and image can contribute to automatic syntactic representation generation.

2 Major Idea of Visually Grounded Neural Syntax Acquisition

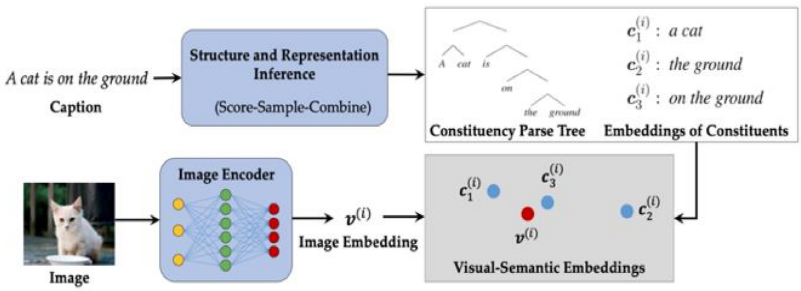

Figure 1:VG-NSL consists of two modules: a textual module for inferring structures and representations for captions and a visual semantic module for matching constituents with images

The purpose of the Visually Grounded Neural Syntax Acquisition model (we'll use VG-NSL for short in the rest of the report) is to acquire syntax of texts, in the form of constituency parsing, by looking at images and reading the captions. It consists of 2 modules, as shown in Figure 1. In the first module, given a caption, VG-NSL builds a latent constituency parse tree using easy-first bottom-up parser, the composition of the tree from a caption of length n consists of n-1 steps, at each step a pair of constituents are sampled from all pairs of current consecutive constituents, pairs are selected by a distribution produced by a score metric that is a parameterized by Θ (the score function is a two-layer feed-forward neural network). Then the pair are combined to form a single new constituent, such that the numbers of constituents are reduced by 1 at each step. the textual representation for the new constituent is defined as the L2 normed sum of the two components. The process of this parsing process is shown in figure 2.

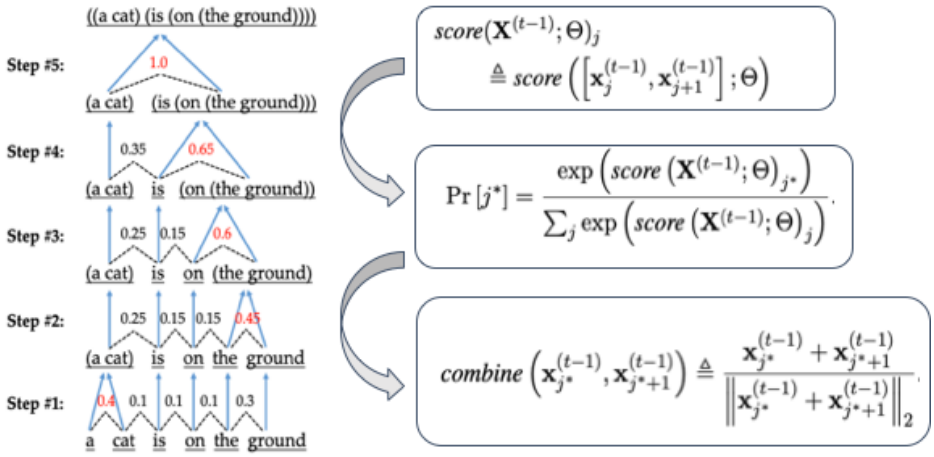

Figure 2: Textual parsing process

In the second module, visual semantic embeddings are used to pair images and text constituents. Through this alignment process, the text constituents are grounded in image representations. The matching score between images and texts are defined as $m(v^{(i)},c_j^{(i)};\Phi) \triangleq cos\ (\Phi v,\ c)$. In training, the visual concreteness of a constituent is defined as $concrete(\boldsymbol{z},\boldsymbol{v}^{(i)}) = \sum_{k\neq i,p}[m(\boldsymbol{z},\boldsymbol{v}^{(i)}) - m(\boldsymbol{c}_p^{(k)},\boldsymbol{v}^{(i)}-\delta')]_+ + \sum_{k\neq i}[m(\boldsymbol{z},\boldsymbol{v}^{(i)}) - m(z,\boldsymbol{v}^{(k)}-\delta')]_+$, where z is the combine function in textual parsing process shown in figure 2. A reward function for a combination of a pair is then defined as $r(\boldsymbol{x}_j^{(t-1)},\boldsymbol{x}_{j+1}^{(t-1)}) = concrete(\boldsymbol{z},\boldsymbol{v}^{(i)})$. To take into account the head initial inductive bias, which is common among English where in verb phrases or prepositional phrases, the verb precedes the complements, an abstract score is defined as $abstract(\boldsymbol{z},\boldsymbol{v}^{(i)}) = \sum_{k\neq i,p}[m(\boldsymbol{c}_p^{(k)},\boldsymbol{v}^{(i)}) - m(\boldsymbol{z},\boldsymbol{v}^{(i)}+\delta')]_+ + \sum_{k\neq i}[m(\boldsymbol{z},\boldsymbol{v}^{(k)}) - m(\boldsymbol{z},\boldsymbol{v}^{(i)}+\delta')]_+$, and a modified reward function as $r'(\boldsymbol{x}_j^{(t-1)},\boldsymbol{x}_{j+1}^{(t-1)}) = \frac{r(\boldsymbol{x}_j^{(t-1)},\boldsymbol{x}_{j+1}^{(t-1)})}{\lambda\cdot abstract(\boldsymbol{x}_{j+1}^{(t-1)},\boldsymbol{v}^{(i)})+1}$. In training, a hinge-based triplet ranking loss is used to optimize textual representations and the visual-semantic embedding space: $L(\Phi;V,C) = \sum_{i,k\neq i,j,l}[m(\boldsymbol{c}_l^{(k)},\boldsymbol{v}^{(i)}) - m(\boldsymbol{c}_j^{(i)},\boldsymbol{v}^{(i)}+\delta)]_+ + \sum_{i,k\neq i,j}[m(\boldsymbol{c}_j^{(i)},\boldsymbol{v}^{(k)}) - m(\boldsymbol{c}_j^{(i)},\boldsymbol{v}^{(i)}+\delta)]_+$.

3 Experiments

The data of this paper comes from MSCOCO data set, which contains 82783 images for training, 1000 for development and another 1000 for testing. Each image is associated with 5 captions. We first established PMI baseline models with different smooth parameters and use the MSCOCO dataset to check if the result we arrive at is comparable with the results shown in the paper.

PMI uses negative pointwise mutual information between adjacent words as syntactic distance, then the tree is formed by recursively splitting the sentence between two adjacent words that have largest syntactic distance. The performance of the PMI baseline model is shown in Table 1. As can be seen from Table 1, the F1 score of the PMI baseline model

ranges from 32.23 to 34.68 under different smooth parameter choices. In the original paper, the author reached an average F1 score of 30.5. This indicates that the result of our baseline model is comparable with those of the original paper and our results are slightly better.

Table 1: PMI baseline model performance

Model	F1 score	Precision	Recall
PMI (SP=0.5)	32.23	29.32	35.78
PMI (SP=1.0)	32.34	29.42	35.9
PMI (SP=1.5)	34.68	31.55	38.50

Note: SP stands for smoothing parameter, SP=0.5 means smoothing parameter is 0.5.

We then replicates the results of the VG-NSL model shown in the original paper, the results of the VG-NSL and VG-NSL+HI (VG-NSL with head initial) models are shown in the first two rows of Table 2, we also included the results of the VG-NSL and VG-NSL+HI of original paper in the parentheses of the first two rows as comparison to our results. As can be seen from the first two rows of Table 2, our results are comparable with those shown in the original paper. This provides partial support to one of the major characteristics of the model stated in the original paper: compared with other baseline models, the performance of VG-NSL models are more robust to different initialization settings. Another important finding is that the F1 score of the VG-NSL model is significantly higher than that of the baseline model, the average F1 score is about twice that of the PMI model, this indicates that the VG-NSL model indeed shows very good performance in terms of F1 score. In addition, the performance of the model is especially good on noun phrases and prepositional phrases which are very common in textual captions.

Table 2: VG-NSL model performance

Model	NP	VP	PP	Overall F1
VG-NSL	**76.3 (79.6)**	28.8 (26.2)	36.3 (42.0)	**51.0 (50.4)**
VG-NSL+HI	**74.2 (74.6)**	26.2 (32.5)	**57.7 (66.5)**	**52.99 (53.3)**

Note: Recall/F1 score of the original paper in the parentheses; NP: recall using noun phrase, VP: recall score using verb phrase, PP: recall using prepositional phrase; HI: head initial.

Overall, we think that VG-NSL model performs really well in terms of F1 score, the model performs especially good on noun phrases and prepositional phrases, which are common in textual captions. The performance of the model is also quite stable in terms of different initializations. In addition, the model uses an unsupervised approach that does not require human-labeled constituency trees. Those characteristics make it a promising future direction in improving the performance of syntactic representations.

4 Exploration and Future Extension Direction

4.1 Revised VG-NSL model with different rewards

As stated in the second part of the report, we use visual concreteness of a constituent to define a reward function. We tried to revise the parameters of the reward function to see if we can improve the results. The reward function stands for the estimator of the gradient. Since this gradient estimator method will likely be diverge, and how will it get closer to optimal depends on the reward function. We tried two methods to revise the reward function: one is to rescale the reward function to see if increasing and decreasing the reward can generate a better result. The other is a thought from head initial bias: it changes the reward by decreasing all reward, but the reward form combining abstract constituency with preceding constituency will decrease more, the reward with next constituency however, will decrease less. We

follow the intuition that we increase all the reward, but the reward from combining abstract with preceding constituency will increase less and with next constituency increasing more. The results are shown in Table 3.

Table 3: Revised VG-NSL model performance

Model	NP	VP	PP	Overall F1
VG-NSL (a=0.8)	71.5 (79.6)	28.3 (26.2)	38.3 (42.0)	49 (50.4)
VG-NSL (a=1.2)	71.6 (79.6)	33.4 (26.2)	35.6 (42.0)	46 (50.4)
VG-NSL (new_re)	21.5 (79.6)	12.3 (26.2)	6.2 (42.0)	15.3(50.4)

Note: Recall/F1 score of the original paper in the parenthese; NP: recall using noun phrase, VP: recall score using verb phrase, PP: recall using prepositional phrase; VG-NSL (a=0.8): VG-NSL with alpha of the reward function equals 0.8; VG-NSL (a=1.2):VG-NSL with alpha of the reward function equals 1.2, VG-NSL(new_re): VG-NSL with new reward function $r'\left(\boldsymbol{x}_j^{(t-1)}, \boldsymbol{x}_{j+1}^{(t-1)}\right) = r\left(\boldsymbol{x}_j^{(t-1)}, \boldsymbol{x}_{j+1}^{(t-1)}\right) \cdot (\lambda \cdot abstract\left(\boldsymbol{x}_{j+1}^{(t-1)}, \boldsymbol{v}^{(i)}\right) + 1)$.

As we can see, rescaling the reward will increase the recall of VP but decrease the recall of NP and PP, and gives similar F1 score. While changing the new reward is kind of unlucky, since all scores decrease a lot. This might due to the reason that increasing all the reward will made the gradient estimator become harder to control, it can become too large in every step. Since this method is already diverge, too large gradient will make it worse. So, rescaling this large gradient to see if this line of thought can provide better results could be a possible direction for future work.

4.2 Reason Behind Data Efficiency

In the original paper, the authors found that "VG-NSL is much more data-efficient than prior work based purely on text (as can be seen from Figure 3), achieving comparable performance to other approaches using only 20% of the training captions". We observed a similar phenomenon during the training process. One possible reason for this is the MSCOCO caption is limited in terms of entities could be involved, relations between entities, so it is likely that with 20% of data, most of entities and relations involved are already seen by the model. Thus, more data is not as much help as we would imagine.

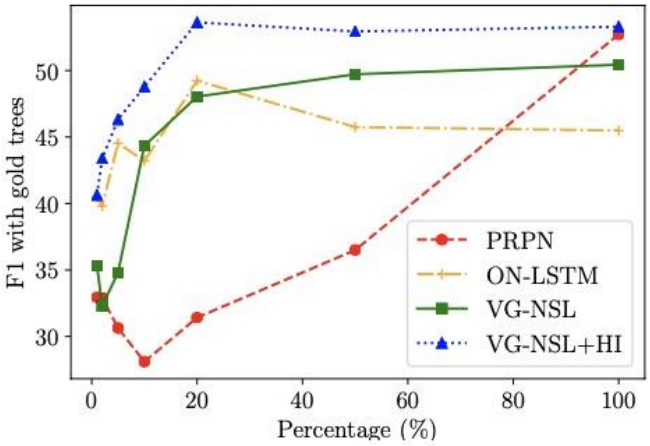

Figure 3: Data efficiency of VG-NSL

To check if this is the case, we randomly sampled 20% of training data, and try to find the percentage of testing data that can be found in that 20% training data. We do this by converting the captions to parse trees and then extract the

Noun Phrase (NP), Verb Phrase (VP), and Prepositional Phrase (PP), we then calculate the percentage of the phrases included in the test data shown up in the training data. The results are shown in table 4.

Table 4: Phrases overlapping rate between 20% training data vs. 100% test data

Num. of experiment	NP	VP	PP
1	0.735	0.392	0.657
2	0.737	0.393	0.666
3	0.731	0.385	0.668
4	0.731	0.393	0.657
5	0.738	0.402	0.665
Average	0.734	0.393	0.662

Note: NP: noun phrase overlapping rate between 20% training data vs. 100% test data, VP: verb phrase overlapping rate between 20% training data vs. 100% test data, PP: prepositional phrase overlapping rate between 20% training data vs. 100% test data.

As shown in the Table 4, the phrase in test data overlap with the 20% of the training data to a very large extent, especially for noun phrase and prepositional phrase, the overlapping rate reaches about 73%! We believe this provides explanation as to why the model reaches achieving comparable performance to other approaches using only 20% of the training captions. Since most of captions consist of phrases that frequently occur, 20% of data is enough to train this model, extra data are unlikely to provide more information to improve the performance of the model. We also calculate this overlapping rate using edit-distance. The reason to do this is that two phrases can be different although they can provide the same meaning. The results are shown in Table 5.

Table 5: Phrases overlapping rate between 20% training data vs. 100% test data (edit-distance)

phrase	NP	VP	PP
similarity	0.937	0.839	0.937

Note: NP: noun phrase overlapping rate between 20% training data vs. 100% test data using edit distance, VP: verb phrase overlapping rate between 20% training data vs. 100% test data using edit distance, PP: prepositional phrase overlapping rate between 20% training data vs. 100% test data using edit distance.

As can be seen from Table 5, the overlapping rate between 20% training data and 100% training data becomes even higher when using edit distance, this provides further evidence as to why 20% data usage could achieve comparable performance with other models.

4.3 Word Embedding Experiment

Word embedding has been proved to be both successful meaning representation for linguistic research: studying word similarities, word meaning changing chronologically, also, for building natural language processing systems like machine translation and information retrieval. Word2vec and Glove were the most popular pre-trained embeddings used by researchers and industry applications, in those pre-trained embeddings, each word has a corresponding vector representing the meaning of that word with a 200/300 (typically) vector into a high dimensional space. With word meaning being embedded into space, measures like the closeness between two words “dog” and “cat” can be computed directly with simple linear algebra such as dot product. But word2vec and Glove suffer from out of vocabulary problem, namely, although those embeddings are trained using a large corpus when used on a specific dataset, there might still

be out of vocabulary words due to miss-spelling or word infections. Some tricks and proposed approaches are to specifically address this issue.

First, we could use a special '<unk>' token and assign it a vector that is computed by a mean of all the embeddings, then for low frequent word, their embeddings are replaced with the '<unk>' embedding. Although this approach performs finely on some tasks like sentiment analysis, obviously, this approach has the potential risk of losing information which could be critical for some words meaning sensitive tasks like machine translation. Another approach is to train char-level embeddings, then any word can be decomposed into a combination of char-level word embeddings, but this method suffers from the fact that by breaking word into chars the embeddings are less informative about word relations but focus more on character associations, so it will have inferior performance when used to build models that addressing tasks that care about word associations and even sentence-level relations compared to word-level embeddings. Then Fasttext embedding using subword embeddings trying to achieve the same effect of giving every word an embedding, it is generally concerned as a good solution for addressing out of vocabulary problem.

Another problem of those one-to-one word embeddings is that each word is only represented by one vector, no matter how context changes, the word only has that meaning. So an intuitive thought would be to produce word embeddings on the fly with respect to the context of a sentence. That is contextual word embeddings, representatives are ELMO, and Bert (Pre-training of Deep Bidirectional Transformers for Language Understanding). Contextual word embeddings have bee generally considered as a milestone in Natural Language Processing, ever since its publishment, nearly every benchmark score is refreshed by models based on contextual word embeddings.

Syntax tree paring relies heavily on the context, for example, to recognize whether a Prepositional phrase is bound to the right constituents, a wide context is needed for the model to make such a decision. We want to investigate if contextual embeddings like BERT could help the existing system from the paper to make better judgment during syntax tree construction utilizing contextual information. And our goal is to better understand the model behavior as well as to achieve a better F1 score.

We did two experiments regarding integrating BERT embedding into the model. The first one is a plain substitution of the embedding layer of text encoder in the original model with a Bert-base-uncased model, substitute the vocabulary and tokenizer into Bert compatible ones. BERT used word piece tokenizer (to divide words into subwords that exist in BERT vocabulary greedily), so we changed the text preprocessing function to be compatible with BertTokenizer.

The result with the first experiment is shown in Table 6. As can be seen from Table 6, the f1 score and recall is very low compared to the original model. The main reason could be BERT use word piece tokenizer, so if we use plain bert vocabulary space, some words will be split into subwords, which could cause problems for our task, for syntactic tree parsing, the atom unit involved should be words, not subwords. This requirement came from the linguistic definition of what is a constituency parsing tree. In a word, by dividing words into subwords, the tree construction phase could involve subwords, and result in invalid trees. This will hurt the F1 score significantly. One solution to the problem might be to re-generate the golden tree in bert vocabulary space so that these two tree formats will be comparable. But of course, from a strict linguistic definition, it would still make no sense.

Table 6: Performance with BERT word embedding

Model	NP	VP	PP	Overall F1
VG-NSL_BERT	20.4	3.5	12.5	12.36

Note: NP: recall using noun phrase, VP: recall score using verb phrase, PP: recall using prepositional phrase.

Therefore, we conducted a second experiment trying to solve the problem by recombining the embeddings of subwords to a single embedding corresponding to the whole world. In order to do that, we tracked those words getting split during tokenizing, and according to that information, we recombined the embeddings in-text encoder. But we encounter some CUDA error during the combination process, perhaps due to how the gradient is computed. We will continue to debug it. But I think it is a valid direction.

5 Conclusion

In this project, our motivation is to investigate if the performance of unsupervised constituency parsing could be improved by inducing information of image and caption associations, and by associations obtained through alternatively optimizing between the alignment of image and caption and constituents score function. We followed the VG-NSL model by Shi et al., 2019. We established the baseline models and the VG-NSL models of the original paper and found similar results as shown in Shi et al.'s paper. VG-NSL model improves the performance of the previous model, its performance is robust in terms of different initializations. Meanwhile, the performance of the model is especially good on noun phrases and prepositional phrases which are very common in textual captions. We further experimented with the model from the following three aspects: changing the parameter of the reward function and changing the composition of the reward function, explore the reason behind the data efficiency of the model and use BERT contextual embedding as word embedding method to see if we could get comparable results. For future work, we want to continue working on the word piece recombination based BERT extension to the original model, also, we want to apply the model to dataset of different languages, and compare it with other methods.

6 Acknowledgement

We want to express our thanks to Professor Chenhao Tan, Kodur Krishna Chaitanya, Han Liu, Siddartha Shankar for your valuable feedbacks and advice.

7 Reference

Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. "Glue: A multi-task benchmark and analysis platform for natural language understanding." *arXiv preprint arXiv:1804.07461* (2018).

Forbes, Maxwell, Ari Holtzman, and Yejin Choi. "Do Neural Language Representations Learn Physical Commonsense?." *arXiv preprint arXiv:1908.02899* (2019).

Shi, Haoyue, Jiayuan Mao, Kevin Gimpel, and Karen Livescu. "Visually Grounded Neural Syntax Acquisition." *arXiv preprint arXiv:1906.02890* (2019).

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft coco: Common objects in context." In *European conference on computer vision*, pp. 740-755. Springer, Cham, 2014.

Plummer, Bryan A., Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. "Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models." In *Proceedings of the IEEE international conference on computer vision*, pp. 2641-2649. 2015.

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