Neural Open Information Extraction with Transformers

Wes Gurnee
Cornell University
Ithaca NY
rwg97@cornell.edu

Ziwei Gu
Cornell University
Ithaca NY
zg48@cornell.edu

Abstract

In this work, we model the Open Information Extraction problem as a sequence to sequence transduction task. We use a learning based approach to train a transformer encoder-decoder architecture to extract relational triples using a large training set bootstrapped from a rule based extractor. We show that our Open IE system significantly outperforms several existing Open IE tools on a large benchmark dataset and is competitive with the state of the art, without the dependencies on other NLP tools.

1 Introduction

Open information extraction is the task of generating a structured representation of the information in text, usually in the form of relational triples (e.g., (LeBron James, plays for, Los Angeles Lakers)). Unlike traditional relation extraction methods, Open IE is not limited to a predefined set of target relations or a taxonomy of entities, but rather extracts all types of relations found in a text. As a result, it enables scaling to large heterogeneous corpora without extensive human involvement.

Open information extraction is important because it enables many down stream NLP applications. The most obvious is in knowledge graph or knowledge base generation. The triples extracted from an Open IE system can be stored as a knowledge graph where the relation is a directed edge that connects the extracted subject to the object. Many other applications like question answering and search systems can also benefit from more structured representations of text.

2 Related Work

Many extraction systems have been proposed since the first Open IE system TEXTRUNNER (Banko et al., 2007). Traditional Open IE systems can be broadly classified into 3 categories - rule-based, clause-based, and learning-based. Rule-based systems like REVERB (Fader et al., 2011) make use of hand-crafted extraction rules involving part-of-speech tags and regular expression patterns. Clause-based systems like CLAUSEIE (Del Corro and Gemulla, 2013) aim to improve the accuracy by incorporating a sentence restructuring stage to locate relations in a set of syntactically simplified clauses. As a result, they can better cope with complex sentences. Learning-based systems like OLLIE (Mausam, 2016), on the other hand, take a self-supervised learning approach that learns patterns based on dependency parse paths. However, all of these systems rely heavily on other NLP tools for pattern matching and may suffer from error propagation.

Recently, supervised neural network models revitalized the field of Open IE. Cui et al. cast Open IE as a sequence generation problem and propose an encoder-decoder framework, which achieved promising results without any hand-crafted patterns (Cui et al., 2018). Arzoo et al., treat Open IE as a sequence labeling problem and use an attention-based recurrent neural network for joint extraction of entity mentions and relations (Katiyar and Cardie, 2017). Stanovsky et al. formulate Open IE as a sequence tagging problem and develop a bi-LSTM transducer to extend deep Semantic Role Labeling models to extract Open IE tuples (Stanovsky et al., 2018). Inspired by these efforts, we investigate a transformer-based sequence-to-sequence approach to obtain binary extractions. To the best of our knowledge, this is the first time such model architecture is used for

Open IE.

3 Approach

We treat the open information extraction problem as a sequence to sequence transduction task inspired by neural machine translation. More specifically, given an input sentence, the task is to produce a sequence containing the subject, object, and predicate delimited by special tokens indicating the type of a the span. For instance, given the input "Barack Obama was President of the Unites States." our objective is to generate the sequence "<arg1>Barack Obama </arg1> <rel>was</rel> <arg2>President of the United States </arg2>."

3.1 Data

To train our seq2seq model, we used the training data made available by (Cui et al., 2018). The data is gathered from Wikipedia dump 20180101 where each example is a sentence with 40 or less words. The labels are gathered from a rule based system OpenIE4[1] where only extractions with confidence greater than .9 are retained to ensure high quality training data. In total, the training set contains 36,247,584 (sentence, tuple) pairs.

We also use the same test set as (Cui et al., 2018) that uses the open information extraction benchmark created by (Stanovsky et al., 2018) that leverages the QA-SRL annotation. The test set contains 3,200 sentences and 10,359 extractions[2].

3.2 Model

Given the recent success of self-attention networks in a variety of sequence processing tasks, we apply the transformer architecture (Vaswani et al., 2017) to this problem. A transformer is composed of a encoder and decoder each of which contain several blocks. An encoder block contains a multi-head attention layer that is added with the input via a residual connection and then normalized. This sequence of vectors is then passed through a fully connected feed forward layer that again has a residual connection to the multi-head attention output. The decoder block looks the same except the first multi-head attention layer is masked so that it can not peek ahead at future inputs, and it has an addition multi-head attention layer that gets some of its inputs from the corresponding encoder block. Because these blocks don't rely on recurrent connections, the transformer can more effectively utilize parallelization and hence is much faster to train.

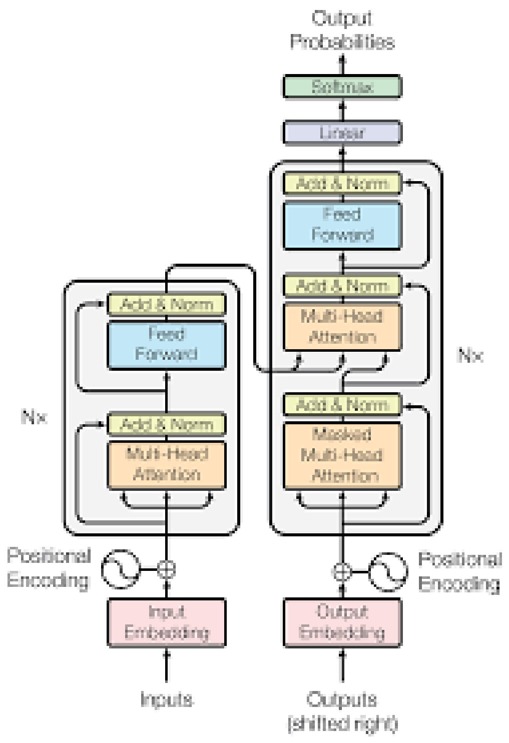

Figure 1: Transformer (Vaswani et al., 2017)

The input sentences are tokenized using a byte-pair tokenizer taken from OpenAI's GPT model (Radford et al., 2018). Byte-pair tokens have had impressive performance in newer models and offer better ways of dealing with rare words and fixed vocabulary sizes. These input embeddings are then added to a positional encoding because without recurrent connections we must provide the model with some information about the ordering of tokens. As in the original paper we use the positional encoding functions:

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{\text{model}}})$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{\text{model}}})$$

The final output of the last decoder block is fed into a final linear layer with a softmax activation function to get a distribution over the output vocabulary.

The main architectural hyperparameters are the number of encoder and decoder blocks, the dimensionality of the input and output d_{model}, the dimensionaility of the feed forward layers, and the

[1] https://github.com/knowitall/openie
[2] https://github.com/gabrielStanovsky/oie-benchmark

number of self attention heads. Our final model has dimension 768, feed-forward dimension 1200, 6 self-attention heads, and 5 encoder and decoder blocks. In total our model has 123,506,387 parameters. See (Vaswani et al., 2017) for more details of the original architecture.

3.3 Training

In addition to using the tokenizer from OpenAI's GPT, we also use the pretrainined token embeddings to warm start our models as these embeddings were the result of a significant amount of training on a bigger corpus. Our source and target vocabularies are about 35000 tokens. Because of memory limits, the training set was split into chunks and each chunk was run for 1-2 epochs. We use the Adam optimizer (Kingma and Ba, 2014) with a decaying learning rate schedule. We apply dropout to linear layers with 0.1 drop probability. Additionally, as the original authors we employ label smoothing with values $\epsilon_{ls} = 0.1$ which makes the model more unsure but has been shown to improve accuracy.

We trained the model over 4 days on a single Nvidia K80 with 12 GB of vRAM. We had a batch size of 9000 tokens where batches were sorted by sentence length to minimize the number of padding tokens to maximize efficiency. In total, our model received processed over a billion tokens during training.

4 Evaluation

Similarly to Cui et al., we used the large gold benchmark corpus introduced by (Stanovsky and Dagan, 2016), with 3200 sentences from Wikipedia and the Wall Street Journal that has 10,359 extractions. We test the performance of different configurations of our model as well as 5 other prominent Open IE systems: ClausIE, OLLIE, OpenIE-4, PropS, and Stanford (See Section 2). For each sentence, our extraction was lined up with multiple gold extractions and the extraction was considered valid if it achieved a high enough lexical coverage of the reference (defined by the percentage of words covered). Subsequently, the precision and recall of our system were analyzed on different confidence thresholds, and the area under the PR curve calculated.

More specifically, recall is defined by

$$\frac{\text{True in what's covered by our extractor}}{\text{True in gold extractions}} \tag{1}$$

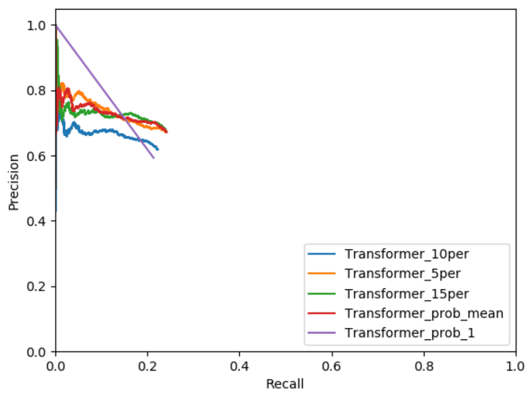

Figure 2: The Precision-Recall (P-R) curve of 4 different configurations of our transformer model

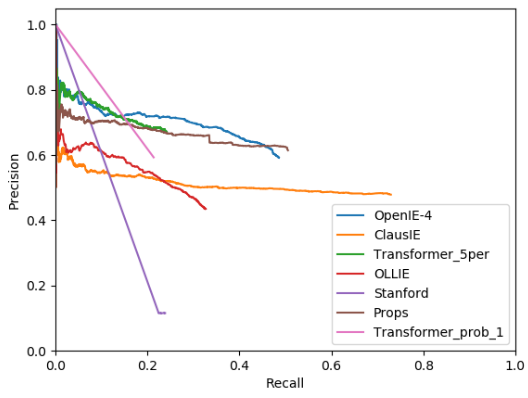

Figure 3: The Precision-Recall (P-R) curve of our transformer model and 6 other Open IE systems

and precision is defined by

$$\frac{\text{True in what's covered by extractor}}{\text{What's covered by extractor}} \tag{2}$$

The confidence scores of our extractions come from the softmax distribution from the transformer decoder. We try different configurations of confidence, assigning to it the mean, minimum, or some percentile of the output probabilities during decoding.

5 Results

It is observed from the Precision-Recall curve (Figure 3) that our transformer model performs significantly better than 4 of the 5 existing rule-based Open IE systems and achieves comparable results with the current best systems OpenIE-4 as well as NeuralOpenIE in (Cui et al., 2018). The Transformer_5per model where confidence is computed by taking the 5th percentile value of softmax probabilities of all tokens in the output achieves the best Area under Precision-Recall Curve (AUC) score among all configurations we tested. It's worth noting that in the Transformer_prob_1 model, we set confidence scores to be 1 for all extractions, similar to Stanford Open IE system (which assigns confidence of 1 to 94% of its extractions). Consequently, the corresponding curve appears linear and has a low precision above 20% recall. None of our models achieves a recall as high as some other systems (noticeably ClausIE, which is best at recall) because our model only produces one extraction per sentence.

We observed many cases in which our transformer is able to correctly identify the boundary of arguments but OpenIE-4 cannot, for instance:

Input	As a group , the team was enshrined into the Basketball Hall of Fame in 1959 .
Gold	the team ||| enshrined ||| into the Basketball Hall of Fame
OpenIE-4	the team ||| was enshrined into ||| the Basketball Hall of Fame
Transformer	the team ||| was enshrined ||| into the basketball hall of fame in 1959

Input	Certain fractional quantum Hall phases appear to have the right properties for building a topological quantum computer .
Gold	certain fractional quantum hall phases ||| have ||| the right properties for building a topological quantum computer .
OpenIE-4	certain fractional quantum hall phases ||| appear ||| to have the right properties for building a topological quantum computer .
Transformer	certain fractional quantum hall phases ||| to have ||| the right properties for building a topological quantum computer .

In the following instance, our transformer is able to correctly replace an ambiguous pronoun in an argument with what it's referring to but OpenIE-4 cannot.

Input	These are known as Porter 's three generic strategies and can be applied to any size or form of business .
Gold	Porter 's three generic strategies ||| can be applied to ||| any size or form of business
OpenIE-4	These ||| can be applied to ||| any size or form of business
Transformer	Porter 's three generic strategies ||| can be applied ||| to any size or form of business

These cases illustrate the generalization capability of our model. Not limited by hand-crafted patterns from other NLP tools, the transformer model reduces the error propagation effect and thus achieves better accuracy.

6 Analysis

6.1 Extraction Quality

In general the extractions are pretty good [3], especially for shorter sentences with simpler clause structures. As the length and complexity of the sentence grows, the extractions are still sensible, but aren't as useful. For instance in a long complicated sentence it is common to segment the primary to-be verb and have the arguments as very long clauses. While still being a somewhat valid extraction, it isn't useful to have clause level spans as entities in a knowledge graph.

Additionally, another pre/post-processing step that would need to be included to make a sentence level extractor more useful is a pronoun resolver. Sentences with extracted pronouns would have to be considered in their context so that the pronouns can be resolved to their proper antecedent.

6.2 Architecture

Because of the cost of training (both in time and dollars) it was not feasible to fully train many different configurations of the model and pick the best one. However, at the onset, we did a few small scale tests to discern what a good architecture might look like. Our experiments suggested having extra heads seemed to help performance without affecting convergence, but having a large feed forward dimension severely hurt convergence. Unfortunately, it is difficult to determine whether the performance differences were caused by a lack of training time for larger models. Additional experiments are need to determine the optimal architecture but recent advances in NLP suggest bigger is better and the size of the data set would support training a larger model. We make our model architecture and weights publicly available. [4]

[3] Test set extractions

[4] https://github.com/wesg52/NOIE

6.3 Runtime

On the test set containing 3200 sentences, run on a Nvidia 1060 GPU with 6GB of vRAM, our model takes about 206 seconds to find all extractions and compute their probability values. Note, the limiting factor for speed is vRAM, as this is what dictates the maximum batch size we can run. Batching significantly reduces runtime as it parrallelizes the model computation. For this reason our model benefits significantly from a more powerful computational setup and has the potential to be faster than any other state of the art approach with sufficient computational power. The downside is that without a GPU extraction can take up to 8400 seconds. Further time reductions can be made with more sophisticated batch decoding and other data transfer optimizations.

6.4 Recall

Despite the superior accuracy of our system, our main weakness is with recall. Because we can only make 1 extraction per sentence, our approach is inherently limited on a data set with over 10000 triples and 3200 sentences. One potential workaround of this limitation, keeping within our general approach, is to use something akin to a modified beam search decoding, basically a parallel hypothesis greedy search. The modification would be to start the search with an initial set of candidates that already contain the first token of candidate arguments. The search would then decode each candidate into its most likely triple. Each fully decoded candidate would be associated with a probability value as before. This value could be used as a threshold, and then we can take the k candidates that achieve this threshold.

7 Conclusion and Future Work

We believe that our results show that a neural approach to open information extraction using transformers is a good approach worthy of further research. With more training time and computation resources our model would have only done better as we could only pass through the full training set once. Additionally, using a bigger model or doing more sophisticated tuning of the architecture parameters would also further improve our results.

Other self-attention neural approaches that would be worth investigating are using just a transformer decoder architecture (Liu et al., 2018) as they are more parameter efficient or fine-tuning a BERT language model (Devlin et al., 2018) to do token classification.

Besides further improving performance, the main problem that our approach has is its inability to scale to multi-sentence extractions or multiple extractions per sentence. This problem is the subject of future research.

Acknowledgments

We thank Professor Claire Cardie for her great class and her advice on our problem formulation and approach. We also appreciate the suggestions and help from Xinya Du on our approaches to the Open IE problem.

References

Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *IJCAI*. volume 7, pages 2670–2676.

Lei Cui, Furu Wei, and Ming Zhou. 2018. Neural open information extraction. *arXiv preprint arXiv:1805.04270* .

Luciano Del Corro and Rainer Gemulla. 2013. Clausie: clause-based open information extraction. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, pages 355–366.

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* .

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, pages 1535–1545.

Arzoo Katiyar and Claire Cardie. 2017. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. pages 917–928.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198* .

Mausam Mausam. 2016. Open information extraction systems and downstream applications. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. AAAI Press, pages 4074–4077.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf* .

Gabriel Stanovsky and Ido Dagan. 2016. Creating a large benchmark for open information extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. pages 2300–2305.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. pages 885–895.

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.

A Contributions

A.1 Joint

We both worked on coming up with the final problem formulation and determining the hyperparameters for the transformer architecture. We also conducted the literature review together to learn about existing approaches and what data sets were available.

A.2 Wes Gurnee

My focus was on the model and training. I adapted an implementation of the transformer to suit our need and integrated the OpenAI tokenizer and token embeddings. I was responsible for preparing the data and training the model on Google Cloud. I implemented our data loading mechanisms as well as our batch decoding and worked on the post-processing.

A.3 Ziwei Gu

My focus was on evaluation and analysis. I applied the trained model to a large benchmark dataset and evaluated the model through samples and PR curves. I also worked on formatting the extractions and some post-processing.

B Examples

Input	Ballast tanks are equipped to change a ship 's trim and modify its stability .
Gold	Ballast tanks ||| equipped to ||| change a ship 's trim and modify its stability
Transformer	Ballast tanks ||| are equipped ||| to change a ship 's trim and modify its stability

Input	In Taiwan , the locals speak a version of the Minnan language which is called Taiwanese .
Gold	the locals ||| speak ||| a version of the Minnan language which is called Taiwanese
Gold	version of the Minnan language ||| called ||| Taiwanese
Transformer	the minnan language ||| is called ||| taiwanese

Input	In 2004 it was expected that redevelopment work in the remaining subway would probably obliterate what remains exist .
Gold	what remains exist ||| would be obliterated
Transformer	redevelopment work in the remaining subway ||| would probably obliterate ||| what remains exist

Input	The town and surrounding villages were hit by two moderate earthquakes within ten years .
Gold	two moderate earthquakes ||| hit ||| the town and surrounding villages
Transformer	the town and surrounding villages ||| were hit ||| by two moderate earthquakes within ten years

Input	There were 22.2 % of families and 23.8 % of the population living below the poverty line , including 15.8 % of under eighteens and 37.5 % of those over 64 .
Gold	22.2 % of families ||| living ||| below the poverty line
Gold	23.8 % of the population ||| living ||| below the poverty line
Transformer	22.2 % of families and 23.8 % of the population ||| living ||| below the poverty line