A Comparison of Transformer Models in Natural Language Processing from a Topological Perspective

Dawei Li

2023.6.14

Abstract

Recently, different models in the Transformer family demonstrate promising results on various NLP tasks. While existing works try to compare and analyze these models from many different perspectives, in this work we do so from a topological view. To do so, we build mapper graphs for word encodings as their topological summaries for each model. Through analysis of coreference mention in a popular TV show dataset, we find the topological summaries between each model are highly consistent with their architectural similarity. We believe our work would provide new insight and perspective for model interpretability of NLP.

1 Introduction

Models in the Transformer [VSP+17] family have demonstrated exceptional performance across a range of Natural Language Processing (NLP) tasks, such as Natural Language Understanding [WSM+18], Text Generation [GSNPB17], and Information Extraction [AGH20]. Different Transformer-based models use distinct architectures; for example, BERT [DCLT18] utilizes an encoder-only architecture, while GPT [RNS+18] utilizes a decoder-only architecture. Moreover, these models adopt different pre-training strategies, such as Mask Language Model for BERT and Auto-regression Generation for GPT. These differences enable each model to excel at specific tasks; for instance, GPT is ideal for text generation, while BERT performs better in language understanding. Having a comprehensive understanding of the differences between each model is crucial since it allows us to select the appropriate model for a particular task, thereby enhancing overall performance. Furthermore, such knowledge offers a novel perspective on the Transformer framework and ways in which it can be improved.

Previous studies have proposed several methods to explore the origin and manifestation of differences between models. In this project, we intend to take a topological perspective and compare them. To achieve this, we apply topological data analysis methods to Transformer models and analyze mapper graph results to gain insights from a topological perspective.

Architecture	Encoder-Only	Decoder-Only	Encoder-Decoder
Representation	BERT	GPT	BART
Pre-training	Mask Language Model	Auto-regressive Generation	Text Denoising
Expertise Area	Understanding	Open Domain Generation	Controlled Generation

Table 1: Comparison of different Transformer architecture

2 Related Work

2.1 Transformer Models Comparison

There are already some works comparing different Transformer models and getting some interesting findings. [Eth19] explore the difference in the contextual word representations of BERT and GPT by calculating the similarity degree of each word in a sentence. There are also some works that focus on different Transformer models' performance in certain tasks [PLC+21, YHZ+20]. Here my idea is to introduce the topological data analysis method into the comparison and see if I can get some novel conclusions based on the result. Table 1 gives a detailed comparison of different Transformer-architecture models.

2.2 Topological Data Analysis in NLP

Recently, some works adopt topological methods in NLP tasks and get promising results. Among them, [VHR+22] introduce words' topological features into Dialogue Term Extraction as additional information. [RDBC22] design a topological algorithm in Word Sense Ambiguous. They analyze every point cloud with and without the word and heuristically use the consistency degree as an ambiguity measurement. On work similar to ours is [RCPW21]. They use topological data analysis to study embeddings of BERT and give some case studies of language elements such as pronouns and syntax.

3 Method

In this section, we give a detailed illustration of how we analyze different Transformer models using mapper graphs. Figure 1 gives an overview pipeline of our analysis process.

3.1 Obtain Word Encodings as Point Clouds

In this section, we first apply some pre-processing to the corpus, including sentence splitting and target word positioning. Then, we extract word encodings from sentence encoding as point clouds for the next step. Different from [RCPW21] which uses activation vectors of the model to construct mapper graphs, we choose to directly use the encoding results output by the model's last layer. That is because many main-stream methods in NLP usually directly adopt word encodings to perform related tasks. So the analysis of word encodings would be more instructive and meaningful to those NLP methods.

To be specific, given a sequence $S = w_1, w_2, ..., w_n$ that consist of n words, we first input them to the Transformer model PLM and get the encoding for the whole sentence $\boldsymbol{H}$:

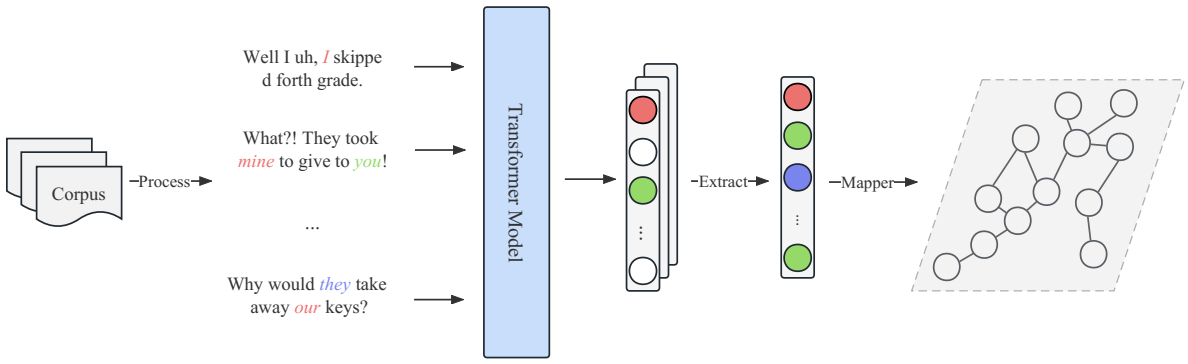


Figure 1: An overview pipeline of our analysis process.

$$\boldsymbol{H} = \text{PLM}(S) \tag{1}$$

Then, we extract the target words' encoding by retrieving the corresponding vectors from $\boldsymbol{H}$:

$$t_1, ...t_m = \text{Extract}(\mathbf{H}) \tag{2}$$

For words that consist of only one token, we use the token's encoding t_i to be the word's representation r_i. For words that consist of more than one token, we adopt a mean pooling to average every token to be the representation of that word:

$$r_i = \text{AVG}(t_p, ...t_q) \tag{3}$$

3.2 Construct Mapper Graphs from Word Encoding

Given a group of word encoding as point clouds, we build their mapper graph to be their topological summary. Each node in the mapper graph represents a cluster of word encoding and an edge that connects two nodes if their corresponding clusters have a nonempty intersection. We follow [RCPW21] to use L2-norm as the filter function and use DBSCAN [EKS+96] as the clustering function. We will give more details about our mapper graph-building process in the experiments section.

4 Experiments

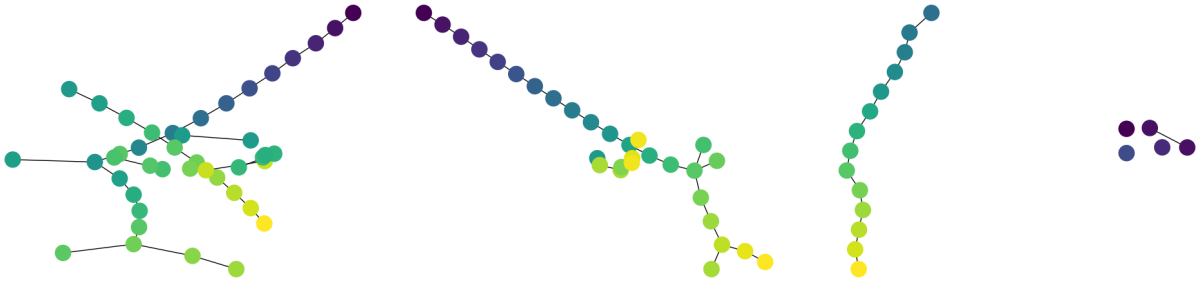
4.1 Setup

For Transformer models, we choose BERT [DCLT18] as a representative for the encoder-only model, GPT [RWC+19] as a representative of the decoder-only model, and BART [LLG+19] as a representative for the encoder-decoder model. We use Pytorch[1] and Transformers[2] to conduct our experiment and analysis.

[1]https://pytorch.org/
[2]https://huggingface.co/docs/Transformers/index

For details of our mapper graph building, we follow [RCPW21] to set n = 70 cover elements with p = 20%. For the clustering algorithm, we set the minimum points per cluster minPts = 5. For the parameter of the DBSCAN algorithm, which defines core points, we use two variations in our experiments. We randomly sample 1,000 encodings for analysis.

As coreference mention (E.g. I, your, his) is one of the most important and basic elements in NLP, we choose to use it to conduct analysis. We use the FRIENDS dataset [SK19] which is a script corpus of the TV show FRIENDS. We extract coreference mentions from each of its sentences using the method we describe in Section 3.1 to extract word encoding as point clouds for analysis.



(a) BERT (b) BART (C) GPT

Figure 2: Topological analysis result using mapper graph

4.2 Results & Analysis

Figure 2 shows the result of the mapper graph of the word encodings. Following [RCPW21], we analyze each of the results from two features: loop and branch. As the figures show, for BERT's result, there are many branches and several loops in the figure. For BART's result, the number of branches decreases obviously, and there is no loop in the figure. We also find that GPT's mapper graph consists of a single limb on the left and several scattered points on the right. That means there is no branch or loop in GPT's mapper result.

We find that topological result interesting because it reflects the architectural similarity of the three models: For BERT and GPT, they consist of the Transformer's encoder and decoder respectively, so their topological summaries show a lot of difference. For BART, it consists of both an encoder and decoder of the Transformer, and the topological summary of it is just like a mixture of the other two models. That indicates different architectures would have an obvious influence on the encoding outputted by models.

5 Conclusion

In this work, we explore a novel topological perspective to compare different Transformer models in NLP. We follow the method proposed by the previous work and use word encoding as point clouds to construct mapper graphs. We choose a basic element in language, coreference mention, as the target to conduct analysis. Our experimental results are highly consistent with the architectural similarity between different models. We believe this would

provide new insight and perspective for works in model interpretability of NLP. In the future, we plan to conduct experiments on more models in NLP and explore other applications of topological data analysis in NLP.

References

[AGH20] Christoph Alt, Aleksandra Gabryszak, and Leonhard Hennig. Tacred revisited: A thorough evaluation of the tacred relation extraction task. *arXiv preprint arXiv:2004.14855*, 2020.

[DCLT18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[EKS+96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[Eth19] Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. *arXiv preprint arXiv:1909.00512*, 2019.

[GSNPB17] Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, 2017.

[LLG+19] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[PLC+21] Beatrice Portelli, Edoardo Lenzi, Emmanuele Chersoni, Giuseppe Serra, and Enrico Santus. Bert prescriptions to avoid unwanted headaches: a comparison of transformer architectures for adverse drug event detection. In *Proceedings of the 16th conference of the European chapter of the Association for Computational Linguistics: main volume*, pages 1740–1747, 2021.

[RCPW21] Archit Rathore, Nithin Chalapathi, Sourabh Palande, and Bei Wang. Topoact: visually exploring the shape of activations in deep learning. In *Computer Graphics Forum*, volume 40, pages 382–397. Wiley Online Library, 2021.

[RDBC22] Michael Rawson, Samuel Dooley, Mithun Bharadwaj, and Rishabh Choudhary. Topological data analysis for word sense disambiguation. *arXiv preprint arXiv:2203.00565*, 2022.

[RNS+18] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[RWC+19] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[SK19] Boaz Shmueli and Lun-Wei Ku. Socialnlp emotionx 2019 challenge overview: Predicting emotions in spoken dialogues and chats. *arXiv preprint arXiv:1909.07734*, 2019.

[VHR+22] Renato Vukovic, Michael Heck, Benjamin Matthias Ruppik, Carel van Niekerk, Marcus Zibrowius, and Milica Gašić. Dialogue term extraction using transfer learning and topological data analysis. *arXiv preprint arXiv:2208.10448*, 2022.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[WSM+18] Alex Wang, 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.

[YHZ+20] Xi Yang, Xing He, Hansi Zhang, Yinghan Ma, Jiang Bian, Yonghui Wu, et al. Measurement of semantic textual similarity in clinical texts: comparison of transformer-based models. *JMIR medical informatics*, 8(11):e19735, 2020.