Legal Knowledge and Information Systems
E. Francesconi et al. (Eds.)

doi:10.3233/FAIA220466

The Effectiveness of Bidirectional Generative Patent Language Models

Jieh-Sheng LEE [1]
National Yang Ming Chiao Tung University School of Law
ORCiD ID: Jieh-Sheng Lee https://orcid.org/0000-0002-0990-6170

Abstract. Generative patent language models can assist humans to write patent text more effectively. The question is how to measure effectiveness from a human-centric perspective and how to improve effectiveness. In this manuscript, a simplified design of the autocomplete function is proposed to increase effectiveness by more than 10%. With the simplified design, the effectiveness of autocomplete can reach more than 60%, which means that more than 60% of keystrokes can be saved by autocomplete. Since writing patent text does not necessarily start from the beginning to the end, a question is whether the generative model can assist a user no matter where to start writing. To answer the question, the generative models in this manuscript are pre-trained with training data in both directions. The generative models become bidirectional. Since text generation is bidirectional, the calculation of autocomplete effectiveness can be bidirectional and starts from anywhere in the text. After thorough experiments, a key finding is that the autocomplete effectiveness of a model for the same text remains similar no matter where the calculation starts. The finding indicates that such bidirectional models can assist a user at a similar level, no matter where the user starts to write.

Keywords. Patent, Natural Language Generation, Natural Language Processing, Deep Learning, Artificial Intelligence

1. Introduction

Large language models have achieved notable success in natural language generation (NLG) tasks in recent years. Until now, very few language models have been dedicated to the patent domain. Furthermore, most language models are autoregressive by predicting the *next* token after having read all the previous ones. Very few language models work backward by predicting the *previous* token. In this manuscript, the author pre-trained and fine-tuned large language models dedicated to the patent domain. The model can predict the previous token, in addition to predicting the next token. Predicting the previous token is implemented by reversing the tokens for both inputs and outputs. The training dataset also contains tokenized sequences in both forward and backward directions. By doing so, the patent-specific language models in this manuscript can generate patent text in both forward and backward directions. The motivation is to make patent text generation flexible because human writing does not necessarily start from the beginning to the

[1] Assistant Professor of Law. Ph.D. in Computer Science. Admitted in New York and passed the USPTO registration exam. Email: jasonlee@nycu.edu.tw

end. Assuming that the thought process in human's mind is a back-and-forth process, a bidirectional generative language model should assist humans more flexibly.

To evaluate the performance of generative patent language models, this manuscript follows the Autocomplete Effectiveness (AE) ratio proposed in [1]. The ratio is used to measure how many keystrokes can be saved for a user if an autocomplete function is provided and based on the generative model. The higher the AE ratio, the more the keystrokes are saved. The contributions of this manuscript include: (1) proposed a simplified version of the autocomplete function to reach higher AE ratios, (2) making the calculation of AE ratios bidirectional, and (3) observed similar AE ratios when using different starting positions of the text for calculation.

2. Related Work

This manuscript is the follow-up work of [1], which proposed the AE ratio to evaluate generative language models. In [1], the patent language model is called PatentGPT-J. The PatentGPT-J models are based on the GPT-J-6B [2] models and pre-trained with a patent corpus from scratch. The use of a Transformer [3] language model for patent text generation was first proposed in [4] by fine-tuning a GPT-2 [5] model with patent corpus. A Transformer model is a deep learning model that adopts the mechanism of self-attention and learns context by tracking relationships in sequential data. The idea of generating patent text backward was introduced in [6]. The research in [6] focuses on controlling patent text generation by structural metadata. However, the effective way to generate text backward and how to measure effectiveness were not addressed in [6]. Except for these works, patent text generation remains a niche research topic less explored.

3. Implementation

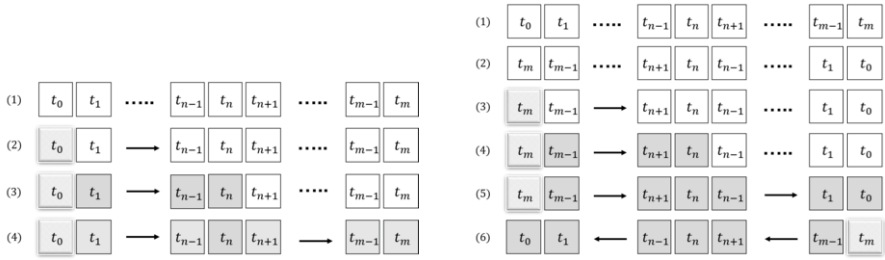
3.1. Simplified Autocomplete Function

The purpose of the AE ratio is to evaluate a language model from a human-centric perspective: how many manual keystrokes can be saved by the autocomplete function based on the generative language model? A higher AE ratio means that the autocomplete function works more effectively and more manual keystrokes are saved. This section describes how the autocomplete function can be simplified to obtain a higher AE ratio. The original AE ratio and the implementations of the PatentGPT-J models are described in [1]. The improvement in AE ratio is achieved by simplifying the user interface (UI). In the conceptual UI design in [1], a user has to press the "downarrow" (↓) key and the "tab" key to complete the autocomplete function. The user selects a preferred token in the top 10 tokens predicted by the model in this way. In this manuscript, a simplified UI is proposed using keys 0∼9 to represent the top 10 tokens. Therefore, the user can select a token by pressing only one key instead of multiple keystrokes. For example, in the previous design in [1], in order to select the 6th token, the user has to press the "downarrow" (↓) key five times and the "tab" key once. Six keystrokes are required. In this manuscript, the user can press the "5" key to select the 6th token of the top 10 tokens. Five keystrokes are saved ("0" key to represent the first token).

3.2. Back and Forth

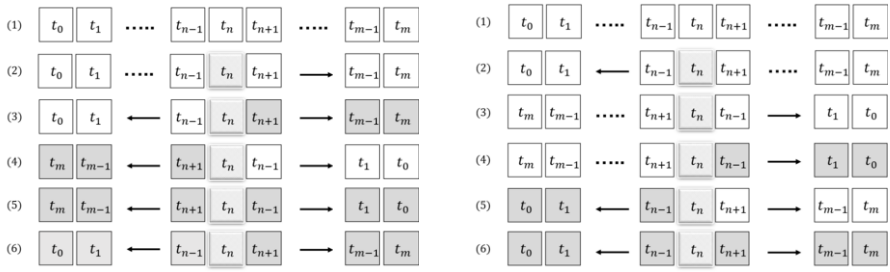
The calculation of the previous AE ratio in [1] is defined as calculating forward to reach the end. Fig. 1a shows how it works. In Fig. 1a, row (1) shows the tokens of the input text after tokenization. Row (2) indicates that the calculation starts from t_0. Row (3) indicates that the calculation moves forward. Row (4) shows that all tokens are calculated after reaching the end. The previous AE ratio in [1] addresses this use case only.

Fig. 1b shows a new use case in this manuscript: calculating backward and starting from the end of the input text. In Fig. 1b, row (1) is the same, showing the tokens of the input text. Row (2) shows the reversed sequence of row (1). Row (3) indicates that the calculation starts from t_m (the end of the original input text). Row (4) indicates that the calculation moves forward for the reversed tokens (backward in effect for the original tokens). Row (5) shows that all tokens are calculated. Row (6) shows the reversed sequence of row (5) to represent the calculation backward from the end of the original input text.



(a) Forward & From the Beginning (b) Backward & From the End

Figure 1. Text Generation



(a) Forward & In the Middle (b) Backward & In the Middle

Figure 2. Text Generation

Fig. 2a and Fig. 2b show two more use cases in this manuscript. In Fig. 2a, row (1) shows the same tokens of the input text. Row (2) indicates that the calculation starts from t_n in the middle (n can be any number between 0 and m) and is about to move forward. Row (3) indicates that the calculation moves forward from t_n, reaches t_m (the end), and is about to move backward. For moving backward, row (3) is reversed as row

(4). The calculation now moves forward starting from t_n again. Given t_m, t_{m-1}, ..., t_{n+1}, t_n as the context, row (5) indicates that the calculation moves forward and reaches t_0 (the beginning). Row (6) shows the reversed sequence of row (5) after moving forward and then backward. For brevity, the description of Fig. 2b is omitted because it is similar to how Fig. 2a works.

3.3. Dataset and Model sizes

The dataset in this manuscript is the USPTO TACD dataset in [1] which includes titles, abstracts, claims, and descriptions in granted patents. The coverage of the dataset ranges from 1976 to 2021 (1976∼2020 for training and 2021 for validation). For further details, please refer to [1]. The model sizes experimented in this manuscript are 6B, 1.6B, and 456M. These three model sizes outperform others (279M, 191M, 128M, and 115M) in [1]. After publication, the pre-trained and fine-tuned models in this manuscript will be released. The model configuration in this manuscript reuses [1]. In terms of training from scratch, this manuscript uses the original GPT-J-6B model and its source code available at [2].

4. Experiments

4.1. Experiment 1

This experiment aims to compare the algorithm in this manuscript with the previous one in [1]. The increase in effectiveness in this manuscript is more than 10%, as shown in the fifth column of Table 1. The first column shows the three models in this experiment: 456M, 1.6B, and 6B. The second column shows the direction for calculating the AE ratio: forward and backward. The third column shows the AE ratios in [1]. The fourth column shows the new AE ratios. The ratio of more than 62% in the fourth column means that more than 62% of keystrokes can be saved by autocomplete in this experiment. The effectiveness of the simplified autocomplete function using keys 0∼9 is validated. The previous combination of the "downarrow" (↓) key and the "tab" key in [1] is eliminated. In Table 1, the "Test Data A" means the patent data used in the first experiment of [1] and reused here. The patent data cover 500 independent claims randomly selected from patents in 2022.

Model	Direction	Previous Ratio (↑)	New Ratio (↑)	Increase (↑)
456M	forward	56.5%	62.7%	10.9%
456M	backward	55.7%	62.1%	11.4%
1.6B	forward	57.0%	63.1%	10.7%
1.6B	backward	56.2%	62.5%	11.2%
6B	forward	57.0%	63.1%	10.7%
6B	backward	56.5%	62.7%	10.9%

Table 1. The improvement of the AE ratio. Target: Test Data A.

4.2. Experiment 2

This experiment implements the mechanism of moving back and forth described in section 3.2. Table 2 shows the experiment results. In this table, the third column "Q1" means the position of the 25%-th token of the input text. The fourth column "Q2" means the position of the 50%-th token of the input text. The fifth column "Q3" means the position of the 75%-th token. The second column defines the direction for calculating the AE ratios. For example, if the input text is tokenized and has 100 tokens, the position "Q1" means that the calculation starts from the 25th token. The "forward" direction in the second column means that the calculation of the AE ratio moves forward to the 26th, 27th, ..., 100th tokens. The 100th token is the end of the input text. After reaching the end, the calculation starts from the 25th token again. The calculation then moves in effect backward to the 24th, 23rd, ..., 1st tokens. The forth-and-back calculation in this example runs over all 100 tokens. In the second column, if the direction is "backward," the calculation will move back first and forth later to run over all 100 tokens. According to Table 2, the AE ratios are similar to one another no matter where the starting position is and no matter which direction to go first. Such a finding indicates that, no matter where a user starts to write, the autocomplete function based on PatentGPT-J models can assist the user to a similar degree and save a similar number of keystrokes. This manuscript hypothesizes that such a finding is not specific to the patent domain and may apply to other generative language models with different training data. This hypothesis can be validated in the future. The "Test Data B" in this experiment contains 1,000 patent claims of CPC Subclass G06N for fine-tuning. These patents are not used in [1].

Model	Direction	Q1	Q2	Q3
456M (fine-tuned)	forward	62.2%	61.7%	61.3%
456M (fine-tuned)	backward	62.4%	62.1%	61.7%
1.6B (fine-tuned)	forward	61.8%	60.9%	60.1%
1.6B (fine-tuned)	backward	61.6%	61.4%	60.9%

Table 2. The AE ratios from different starting positions. Target: Test Data B.

5. Conclusion

Generative language models have great potential to assist humans in writing patent text more effectively. In this manuscript, the way to measure effectiveness is to calculate the ratio of keystrokes saved by the autocomplete function based on model predictions. A higher ratio means more saved keystrokes and fewer manual typing. The ratio was proposed in previous work as the AE ratio. After using a simplified design in this manuscript, the AE ratio increases by more than 10% and reached more than 60%. This means that more than 60% of keystrokes can be saved by the autocomplete function. Furthermore, the models are bidirectional and make it possible to calculate the AE ratio in both directions. The calculation can start anywhere in the text. A key finding is that the AE ratio for the same text remains similar regardless of where the calculation starts. This finding indicates that such bidirectional models can assist a user at a similar level, no matter where

the user starts to write. In addition to the patent domain, the research in this manuscript can be applied to other legal domains in the future because the Transformer architecture in generative models is domain-agnostic.

6. Acknowledgments

The research reported in this manuscript has been funded by the Ministry of Science and Technology (MOST) in Taiwan (Project ID: 111-2222-E-A49-005). In addition, the author would like to thank TensorFlow Research Cloud (TRC) greatly for providing powerful computational resources to make things happen.

References

[1] Lee JS. Evaluating Generative Patent Language Models. arXiv preprint arXiv:220614578. 2022. Available from: https://arxiv.org/abs/2206.14578.

[2] Wang B, Komatsuzaki A. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model; 2021. https://github.com/kingoflolz/mesh-transformer-jax.

[3] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is All You Need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS'17. Red Hook, NY, USA: Curran Associates Inc.; 2017. p. 6000–6010.

[4] Lee JS, Hsiang J. Patent claim generation by fine-tuning OpenAI GPT-2. World Patent Information. 2020;62:101983. Available from: https://www.sciencedirect.com/science/article/pii/S0172219019300766.

[5] Radrof A, Wu J, Child R, Luan D, Amodei D, Sutskever I. Language Models are Unsupervised Multitask Learners; 2018.

[6] Lee JS. Controlling Patent Text Generation by Structural Metadata. In: Proceedings of the 29th ACM International Conference on Information and Knowledge Management. CIKM '20. New York, NY, USA: Association for Computing Machinery; 2020. p. 3241–3244. Available from: https://doi.org/10.1145/3340531.3418503.

[7] Lee JS. PatentTransformer: A framework for personalized patent claim generation. In: Seventh Doctoral Consortium of JURIX 2019. Madrid, Spain; 2019. Available from: http://ceur-ws.org/Vol-2598/paper-06.pdf.

[8] Lee JS, Hsiang J. Prior Art Search and Reranking for Generated Patent Text. In: Proceedings of the 2nd Workshop on Patent Text Mining and Semantic Technologies (PatentSemTech) 2021, co-located with the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021). PatentSemTech 2021; 2021. p. 18-24. Available from: http://ceur-ws.org/Vol-2909/paper2.pdf.

[9] EleutherAI. pile-uspto; 2020. https://github.com/EleutherAI/pile-uspto.

[10] BigScienceCorpus. pile-uspto; 2022. https://huggingface.co/spaces/bigscience/BigScienceCorpus?source=the_pile_uspto.

[11] Brown T, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, et al. Language Models are Few-Shot Learners. In: Larochelle H, Ranzato M, Hadsell R, Balcan MF, Lin H, editors. Advances in Neural Information Processing Systems. vol. 33. Curran Associates, Inc.; 2020. p. 1877-901. Available from: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[12] Lee JS. PatentGPT-J Repository; 2022. https://github.com/jiehsheng/PatentGPT-J.