Improving Text Auto-Completion with Next Phrase Prediction

Dong-Ho Lee[1,2], Zhiqiang Hu[3]*, Roy Ka-Wei Lee[3],
[1]Department of Computer Science, University of Southern California
[2]Upstage AI Research, Republic of Korea
[3]Singapore University of Technology and Design
`dongho.lee@usc.edu, zhiqiang_hu@mymail.sutd.edu.sg,`
`roy_lee@sutd.edu.sg`

Abstract

Language models such as GPT-2 have performed well on constructing syntactically sound sentences for text auto-completion task. However, such models often require considerable training effort to adapt to specific writing domains (e.g., medical). In this paper, we propose an intermediate training strategy to enhance pre-trained language models' performance in the text auto-completion task and fastly adapt them to specific domains. Our strategy includes a novel self-supervised training objective called *Next Phrase Prediction (NPP)*, which encourages a language model to complete the partial query with enriched phrases and eventually improve the model's text auto-completion performance. Preliminary experiments have shown that our approach is able to outperform the baselines in auto-completion for email and academic-writing domains.

1 Introduction

Natural language interface (NLI) applications such as Personal assistants (e.g., Amazon Alexa, Apple Siri, Google Assistant, and Microsoft Cortana) and search engines (e.g., Google) have become an integral part of our everyday life. Among the many features in NLI applications, *text auto-completion*, which aims to suggest words, phrases, and sentences that complete the user's textual input, is a common, but key feature. *Smart reply* (Kannan et al., 2016) and *Smart compose* (Chen et al., 2019) are two recent works that provide contextual assistance to aid users in completing everyday text such as emails, search engine inputs, etc.

While recent advances in deep neural models have shown impressive performance on the text auto-completion task, these models generally require a large amount of everyday text and huge amount of computing power for training to generate adequate suggestions (Chen et al., 2019). The

Test Perplexity	Email	Academic Writing
Bi-LSTM	1.88 ±0.05	3.17 ±0.03

Table 1: **Perplexity Comparison of Bi-LSTM.** Train Bi-LSTM for the language modeling with the same amounts (100K) of training instances for each domain. Perplexity of Academic writing domain is almost double of emails.

Computer Science related article

GPT-2 Suggestions

Figure 1: **Comparison of generated outputs.** GPT-2 can generate syntactically sound, and semantically general sentence from partial query. However, it still needs to be fine-tuned a lot to generate semantically expert domain (e.g. Computer Science) focused sentence.

challenge is compounded when we perform auto-completion in specific domains such as academic writing, which requires a large training corpus for specific expertise. Table 1 illustrates the difficulty in domain-specific auto-completion with the same amount of supervisions.

A potential solution to address the challenges in text auto-completion is exploiting the Decoder-only Transformer model such as GPT-2 (Radford et al., 2019). The model performs well on constructing syntactically sound sentences from a partial query. However, GPT-2 requires a huge fine-tuning effort to construct sentences of expert domains. Figure 1 shows an example of GPT-2 auto-completion suggestions for computer science domain sentences before fine-tuning. Recently, text-to-text transformers such as BART (Lewis et al., 2020) and T5 (Raffel et al., 2020) have demonstrated great potential in natural language generation (NLG) tasks by using

4434

masked-span infilling as a pre-training objective. However, similar to GPT-2, these models also require huge fine-tuning efforts to perform domain-specific text auto-completion.

This paper aims to address this research gap by proposing an intermediate training strategy (Pruksachatkun et al., 2020; Zhou et al., 2021), which incrementally trains a pre-trained text-to-text transformer to provide better auto-completion suggestions and fastly adapt to the expert domain during fine-tuning. As shown in Figure 2, the core of our intermediate training strategy is a simple self-supervised objective called *Next Phrase Prediction (NPP)*, which has two major steps: Phrase Extraction (Section 3.1) and Generative Question Answering (Section 3.2). The first stage extracts qualitative phrases by constituency parsing. By exploiting constituency parsing, the framework is able to utilize the complete phrase, not just a fraction of the sentence. Next, the pre-trained language model is guided to choose the correct next phrase among other phrases of the same type (e.g., noun phrase, verb phrase, etc.) in the sentence. For example, the sentence *"She bought a top and bottom from that strange little shop."* has two noun phrases *"a top and bottom"* and *"that strange little shop"*. If the partial query is *"She bought"*, the model is guided choose the proper complete noun phrase *"a top and bottom"* for its next phrase.

To the best of our knowledge, this is the first work that proposed an intermediate training strategy for improving language models' performance on the text auto-completion task. Through extensive experiments, we demonstrated that our proposed approach could improve the text-to-text transformer's performance on auto-completion task and fastly adapt to expert domain of text auto-completion.

2 Overview

In this section, we first formalize the auto completion problem, and then introduce the workflow of our intermediate training strategy.

2.1 Problem Statement

Given partial query $\mathbf{p} = [s^{(1)}, s^{(2)}, \cdots, s^{(n)}]$, an auto completion returns $\mathbf{q} = [t^{(1)}, t^{(2)}, \cdots, t^{(n)}]$, where $\mathbf{q}$ is a syntactic and semantic extension of $\mathbf{p}$. Specifically, every tokens of $\mathbf{p}$ is a prefix of $\mathbf{q}$, and every tokens of $\mathbf{q}$ is a suffix of $\mathbf{p}$; $[\mathbf{p}; \mathbf{q}]$ is a full sentence. We evaluate the auto completion model's
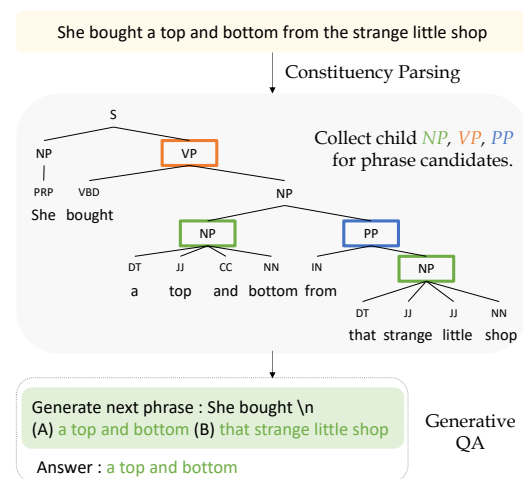

Figure 2: **Overview of next phrase prediction.** From the constituent tree, we retrieve the child phrases and group them according to their types (i.e., noun phrase (NP), verb phrase (VP), preposition phrase (PP).) Next, we randomly select a group that contains more than two phrases. Finally, We construct a generative QA style instance, where the phrases in the group are options to be selected as the correct next phrase for the input phrase.

performance on two attributes: (a) the soundness of $\mathbf{q}$, and (b) the semantic similarity of $\mathbf{q}$ with the ground truth.

2.2 Workflow

The workflow consists of two main steps, starting with a pre-trained T5 model: (i) applying the proposed self-supervised objective *NPP* for intermediate-task training, and (ii) fine-tuning on target auto-completion task.

3 Next Phrase Prediction

The key idea of the *next phrase prediction (NPP)* objective is to train a text-to-text transformer to complete the partial query with adequate phrases. The underlying intuition of our proposed approach is as follows: (1) Phrases tend to express meaning beyond simple word concatenation. For example, noun phrase such as *"Recurrent Neural Network"* is constructed by three different words (*"Recurrent"*, *"Neural"*, *"Network"*), where each word has its own meaning. (2) Common phrases tend to be used on their own in the text. For instance, the prepositional phrase such as *"in this paper"* frequently appears in academic writing domain. Unlike existing language models that are trained to neglect such characteristics of phrases and predict the next word or span of the text, text auto-completion can be improved by performing phrase-level text com-

pletion as an intermediate training strategy in an effort to make the most of the phrase. Specifically, *NPP* involves two main steps: (i) Phrase Extraction, and (ii) Generative Question Answering (QA).

3.1 Phrase Extraction

We first begin by extracting phrases using constituency parsing to retrieve qualitative phrases. Given an input $\mathbf{x} = [x_1, x_2, \ldots, x_n]$, we first conduct constituency parsing using AllenNLP (Gardner et al., 2017) and extract the *Noun Phrase* (NP), *Verb Phrase* (VP), and *Prepositional Phrase* (PP). The extracted phrases are grouped into sets according to their types, denoted as $\mathcal{S}_{vp}, \mathcal{S}_{np}$, and $\mathcal{S}_{pp}$, respectively: $\mathcal{S}_{vp} = [vp_1, vp_2, \ldots, vp_q], \mathcal{S}_{np} = [np_1, np_2 \ldots, np_q]$, and $\mathcal{S}_{pp} = [pp_1, pp_2 \ldots, pp_q]$. For each phrase, we only keep the node that does not have a child node of the same phrase type. For example, the sentence *"She wants to eat pie."* has three VPs as follows:

(1) *wants to eat pie (VP) → wants (VBZ) to eat pie (VP)*
(2) *to eat pie (VP) → to (TO) eat pie (VP)*
(3) *eat pie (VP)*

To construct $\mathcal{S}_{vp}$ for this sentence, we only consider *"eat pie"* as vp_i to avoid word overlap between phrases.

3.2 Generative QA

After retrieving the phrases, we train the language model to predict the correct next phrase in a generative QA task setting (Khashabi et al., 2020). Specifically, from $\mathcal{S}_{vp}, \mathcal{S}_{np}$, and $\mathcal{S}_{pp}$, we randomly choose a set $\mathcal{S}$ that has more than two phrases. To formulate the Generative QA task with the selected $\mathcal{S}$, here we present both the question and answer: If the answer is a randomly chosen phrase p from $\mathcal{S}$, then the question is composed of partial query $\mathbf{p}$ in which the chosen phrase p is an extension of $\mathbf{p}$ and all phrases in $\mathcal{S}$ as answer choices. The model is trained to output the correct phrase p, given partial query $\mathbf{p}$ and answer choices $\mathcal{S}$. Figure 2 shows a real example of this format by choosing $\mathcal{S}_{np}$ as $\mathcal{S}$, *"a top and bottom"* as p, and *"She bought"* as $\mathbf{p}$.

4 Experiments

4.1 Details for intermediate training

We train a pre-trained T5-base model with *NPP*. We randomly sample 1M sentences from the English Wikipedia corpus[1], which is used for pre-training BERT and its variants, as the source data

---

[1]https://dumps.wikimedia.org/enwiki/latest/

Dataset	Train	Dev	Test
Emails	156,998	13,474	15,030
Academics	161,885	20,206	19,953

Table 2: **Statistics of datasets.**

for *NPP*. The corpus has about 1.2B tokens, which is considerably less than the 34B token used in T5, and 10B tokens used in GPT2[2].

4.2 Target Dataset

To show the effectiveness of our proposed method, we utilize two domains of text corpus to create the text auto-completion datasets:

- **Email**: We utilize Enron email corpus [3] for general domain which is written in English collected from internal communication within a large business organization.

- **Academic writing**: We collect the abstracts of academic articles from ArnetMiner (Tang et al., 2008). The articles are written in English and mainly from the Computer Science domain, which are extracted from DBLP [4], ACM [5], etc.

Table 2 summarizes the statistics of the datasets used in our experiments. For data processing, we first extract the sentences from these text corpus. For each sentence, we split into pairs $(\mathbf{p}, \mathbf{q})$ by all word points. We consider $\mathbf{p}$ as partial phrase query to predict completion of the remaining phrase $\mathbf{q}$ in the sentence. Note that the $(\mathbf{p}, \mathbf{q})$ formulation is used in fine-tuning the base models.

4.3 Base models

We compare our proposed approach with other pre-trained language generation models. We fine-tuned the following models on our training data in a sequence-to-sequence format: (1) **GPT-2** (Radford et al., 2019) is the pre-trained GPT-2 large model, which has 774M parameters. For fine-tuning, we condition the model on the format $\mathbf{p} = \mathbf{q}$. For inference, we sample from the fine-tuned GPT-2 model after a prompt of the partial query $\mathbf{p}$ with beam search, and cleaning the samples by post-processing. Then, we use the first sample as the output sentence. (2) **T5** (Raffel et al., 2020) is the pre-trained T5-base model, which has 220M parameters. For fine-tuning, we prepend the prefix:

---

[2]Assuming the average token size is four characters.
[3]https://www.cs.cmu.edu/~/enron/
[4]https://dblp.org/
[5]https://www.acm.org/

Model / Metrics	Emails				Academic Writing			
	BLEU-4	METEOR	CIDEr	SPICE	BLEU-4	METEOR	CIDEr	SPICE
GPT-2 (Radford et al., 2019)	1.1	6.6	26.4	3.3	0.6	6.0	23.6	2.6
T5 (Raffel et al., 2020)	2.8	6.8	39.8	4.2	2.2	7.5	50.3	3.9
NSP+T5	3.0	6.9	41.1	4.4	2.3	7.5	51.1	4.0
NPP+T5 (Ours)	**3.2**	**7.1**	**43.0**	**4.5**	**2.5**	**7.8**	**53.5**	**4.2**

Table 3: **Experimental Results**. The first group of models are baselines which are not intermediately trained. Last group of models are intermediately-trained with different objectives. Best models are bold within each metric.

Partial Query	Original	T5	NPP+T5
Building large OCR databases is a time vpi is part of the ieee programming a connection between the kalman appendix provides a complete listing automatic target	consuming and tedious work . language interface standard . filter is developed . of code for the systems . recognition is an important task .	challenging . system . et al . of the apl libraries . selection is based on a set of criteria .	consuming task . language . filter is established . of the tools and techniques used in this paper . detection is a key feature of this approach .

Table 4: **Generated Examples of Academic Writing**. For the same partial queries from academic writing dataset, we compare the generated completions between T5 and NPP+T5. Underlines are overlap words between the original completion and generated completions.

"generate next phrase:" to partial query **p** and feed into the model to generate completion **q**. (3) **NSP+T5** is intermediately-trained based on T5-Base using Next Sentence Prediction (NSP), which is used in BERT (Devlin et al., 2019) pre-training. (4) **NPP+T5** is intermediately-trained based on T5-base using Next Phrase Prediction (NPP), which is our proposed objective.

4.4 Evaluation Metrics

To evaluate the syntactic and semantic soundness of generated sentences, we exploit several widely used automatic metrics to assess the performance, such as BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), CIDEr (Vedantam et al., 2015), and SPICE (Anderson et al., 2016). These metrics evaluate whether the model is able to generate semantically expert domain focused sentence by measuring surface similarities and associations between system generations and original text.

4.5 Experimental Results

Table 3 shows the experimental results of text auto-completion on the email and academic writing datasets. We observed that the model intermediately trained with our objective outperforms the base models on both datasets.

Specifically, our approach, **NPP+T5**, outperforms **NSP+T5** by a margin from 0.2 to 0.3 BLEU/METEOR/SPICE score, suggesting that predicting the next phrase is more effective than predicting next sentence in text auto-completion task. Moreover, we also observe that **NPP+T5** outper-

forms **GPT-2** even though the number of parameters in **NPP+T5** is less than half of **GPT-2**. The experimental results demonstrated the flexibility of our proposed approach, which can serve as "plug-and-play" for any text-to-text transformer models and enhance their performance in the text auto-completion task.

Table 4 shows the comparison of generated suggestions for the same partial query between **T5** and **NPP+T5**. We can observe that the completions by **NPP+T5** are generally more acceptable in terms of semantic similarity between generated completions and original text.

5 Conclusion

In this paper, we propose a novel intermediate training strategy that encourages the model to complete the partial query with enriched phrases and eventually improving the performance of the text auto-completion system. Our proposed approach enhances state-of-the-art language model's performance by intermediately training it with our next phrase prediction self-supervised objective. Preliminary experiments have shown that our approach is able to outperform the baselines in auto-completion for email and academic-writing domains with only around 1.2B tokens of training. For future work, we aim to experiment our proposed approach on text auto-completion in more writing domains and develop a demonstration system to better showcase our approach in text auto-completion.

References

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In *ECCV*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Mia Xu Chen, Benjamin N Lee, Gagan Bansal, Yuan Cao, Shuyuan Zhang, Justin Lu, Jackie Tsay, Yinan Wang, Andrew M Dai, Zhifeng Chen, et al. 2019. Gmail smart compose: Real-time assisted writing. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2287–2295.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, Laszlo Lukacs, Marina Ganea, Peter Young, et al. 2016. Smart reply: Automated response suggestion for email. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 955–964.

D. Khashabi, S. Min, T. Khot, A. Sabhwaral, O. Tafjord, P. Clark, and H. Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. *EMNLP - findings*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

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 of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5231–5247, Online. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Wangchunshu Zhou, Dong-Ho Lee, Ravi Kiran Selvam, Seyeon Lee, Bill Yuchen Lin, and Xiang Ren. 2021. Pre-training text-to-text transformers for concept-centric common sense. In *ICLR*.